

Command and Control Toolkit™

User's Manual

2.0

Command and Control Toolkit™

User's Manual

September 2002

Copyright © 2001 Command and Control Technologies Corp.

ALL RIGHTS RESERVED. This document contains proprietary and confidential information of Command and Control Technologies Corporation. No part of this publication may be disclosed to third parties, copied, or duplicated in any form without the prior written permission of Command and Control Technologies Corporation.

Command and Control Toolkit, T-Zero and RangeNet are trademarks of Command and Control Technologies Corporation. All other trademarks used in this document are the property of their respective owners.

Contributors

Written by Rodney Davis and Eric Sorton

Edited by Kevin Brown

Production by April Dowden, Vicki Gardner, and Patti Schill

Engineering contributions by Greg Hupf, Frank Noble, and John Ward

Corporate Headquarters

Command and Control Technologies Corp.
1425 Chaffee Drive, Suite 1
Titusville, FL 32780 USA
Tel: 321.264.1193
Fax: 321.383.5096
Web: <http://www.cctcorp.com>

Technical Support

Tel: 321.264.1193
Email: support@cctcorp.com
Web: <http://www.cctcorp.com/support>

LIMITED WARRANTY

SOFTWARE. Command and Control Technologies Corporation (CCT) warrants for a period of THIRTY (30) DAYS from the date of purchase that the Command and Control Toolkit Version 2.0 (SOFTWARE) will execute its programming instructions as specified in the user documentation when properly installed. Due to the complex nature of computer software, CCT does not warrant that the operation of the SOFTWARE will be uninterrupted or error free.

MEDIA. CCT warrants the media upon which this SOFTWARE is recorded to be free of defects in materials and workmanship under normal use for a period of THIRTY (30) DAYS from the date of purchase.

REMEDIES. In the event this SOFTWARE fails to execute its programming instructions, or if any media proves to be defective during the warranty period, your remedy shall be to return the media to CCT for replacement. Should CCT be unable to replace the media within a reasonable time, your alternative remedy shall be a refund of the purchase price upon return of the product and all copies.

WARRANTY VOID. This limited warranty is void if the failure results from abuse, misapplication, or user modification of the software other than that provided for in the user documentation.

NOTICE OF WARRANTY CLAIMS. You must notify CCT in writing of any warranty claim prior to the expiration of the warranty period.

NO OTHER WARRANTIES. To the maximum extent permitted by applicable law, CCT disclaims all other warranties, either express or implied, including but not limited to implied warranties of merchantability and fitness for a particular purpose, with respect to the SOFTWARE. This limited warranty gives you specific legal rights. Some jurisdictions do not allow limitations on how long an implied warranty lasts, so the above limitations or exclusion may not apply to you.

NO LIABILITY FOR CONSEQUENTIAL DAMAGES. To the maximum extent permitted by applicable law, in no event shall CCT be liable for any lost revenues or profits, loss of data or other special, indirect, incidental or consequential damages, even if CCT has been advised of the possibility of such damages. Because some jurisdictions do not allow the exclusion or limitations of liability for consequential or incidental damages, the above limitation may not apply to you.

JURISDICTION. This license is governed by the Laws of the United States.

U.S. GOVERNMENT RESTRICTED RIGHTS. The SOFTWARE and the documentation are "commercial items" as that term is defined in 48 C.F.R. 2.101 consisting of "commercial computer software" and "commercial computer software documentation" as such terms are used in 48 C.F.R. 12.212. Consistent with 48 C.F.R. 12.212 and 48 C.F.R. 227.7202-1 through 227.7202-4, if the licensee hereunder is the U.S. Government or any agency or department thereof, the SOFTWARE and the documentation are licensed hereunder (i) only as a commercial items, and (ii) with only those rights as granted to all other end users pursuant to the terms and conditions hereof.

CCTK User's Manual

MNL-CCTK-User-120302.doc

September 27, 2001

TABLE OF CONTENTS

1	INTRODUCTION	3
1.1	MANUAL ORGANIZATION	3
1.2	CCTK DOCUMENTATION GUIDE	3
1.3	REFERENCE DOCUMENTATION	4
1.4	CCT HELP DESK	5
2	CCTK THEORY OF OPERATIONS	6
2.1	THEORY OF OPERATION OVERVIEW	6
2.1.1	Quality Attributes and Features	8
2.1.2	Operations Process Flow	9
2.2	PHYSICAL ARCHITECTURE	10
2.2.1	Architecture Layers	12
2.2.2	Platforms	13
2.2.3	External Interfaces	13
2.2.4	Networks	14
2.2.5	Performance Considerations	15
2.3	SOFTWARE ARCHITECTURE	19
2.3.1	Real-time Kernel Segment	20
2.3.2	External Interface Segment	27
2.3.3	Domain Application Segment	28
2.3.4	Support Utilities Segment	33
3	CCTK REAL-TIME OPERATION	42
3.1	CCTK CLIENT	42
3.1.1	Overview	42
3.1.2	Obtaining Help	43
3.1.3	Opening a Project	44
3.1.4	Closing a Project	47
3.1.5	Starting a Project	47
3.1.6	Stopping a Project	49
3.1.7	Exiting CCTK Client	50
3.1.8	Display Tree	50
3.1.9	Multiple Document Interface Window	51
3.1.10	Status Bar	52
3.1.11	Individual Displays	52
3.2	SYSTEM MESSAGE DISPLAY	58
3.2.1	Overview	58
3.2.2	Clearing Messages from a Pane	59
3.2.3	Scroll Lock	59
3.2.4	Filters	60
3.2.5	Working with Panes	62
3.2.6	A Practical Example	63
3.3	FD SELECTION	63
3.3.1	Overview	64
3.3.2	Selecting/Deselecting FD's	64
3.3.3	Filtering FD's	65

3.3.4	Selecting/Deselecting All FD's	65
3.3.5	Sorting FD's	66
3.4	REAL-TIME MEASUREMENT MONITORING	66
3.4.1	Selecting FD's to View	67
3.4.2	Removing FD's	68
3.4.3	Options	68
3.4.4	Saving and Loading Measurement Lists	69
3.4.5	Keyboard Accelerators	69
3.5	DATA RETRIEVALS	69
3.5.1	Current Project	70
3.5.2	Retrieval Type	71
3.5.3	Retrieval Time	71
3.5.4	Output File	73
3.5.5	System Messages	73
3.5.6	FD List	73
3.5.7	Measurement Details	74
3.5.8	Retrieval Execution	76
3.5.9	Saving and Loading Retrieval Parameter Files	77
3.5.10	Batch Reports	77
3.6	VIRTUAL STRIP CHARTS	77
4	CREATING DYNAMIC VISUALIZATION DISPLAYS.....	80
4.1	USING THE GLG ENVIRONMENT TO CREATE A DISPLAY	80
4.2	BINDING MEASUREMENT DATA TO A DISPLAY	81
4.3	BINDING MEASUREMENT STATUS TO A DISPLAY	82
4.4	CONNECTING A DISPLAY TO CCTK	85
	GLOSSARY	86
	INDEX	93

1 INTRODUCTION

The Command and Control Toolkit™ product is a data acquisition and commanding software package that can be tailored to a wide range of command and control applications. The product, also known as CCTK, provides a customizable client/server environment designed to meet the performance requirements of mission critical command and control operations. The CCTK client/server environment can be easily expanded from a single display console to any number of networked display consoles.

The CCTK documentation set provides a comprehensive description of how to install, configure, and use the CCTK software. This documentation, together with the reference documents listed in each manual, contains the information you will need to apply CCTK and any optional modules you have purchased to your command and control operation.

1.1 Manual Organization

This document describes the general user operation of the CCTK. It is intended to be a guide for new users of the software. It describes the CCTK theory of operation and real-time operational user interface.

Section 2 discusses the CCTK theory of operation to explain each major function of a project or mission. The overall CCTK architecture and each of the major components are also described.

Section 3 discusses user operations details from the perspective of the graphical user interface. All standard CCTK graphical tools and utilities are described for the novice CCTK user.

Section 4 addresses the CCTK process for creating dynamic visual displays.

A glossary of commonly used terms, phrases, and acronyms is included at the end of the manual.

1.2 CCTK Documentation Guide

This manual is one of a set of five manuals included in the CCTK product suite, illustrated in Figure 1-1 below. The manuals are designed to address the needs of particular classes of CCTK users such as installers, administrators, and operators.

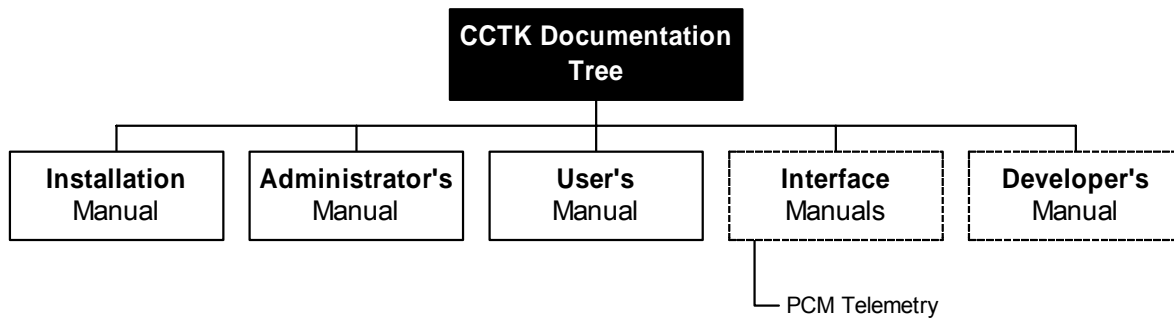


Figure 1-1. CCTK Documentation Tree. Dashed boxes indicate manuals for optional CCTK software products.

Release notes provide supplemental information to this manual. Please refer to the RELEASE_NOTES text file in the root directory of the CCTK CD.

- ***CCTK Installation Manual*** – Describes the steps needed to install CCTK client and CCTK server software from the CCTK CD. Installation on Microsoft Windows® and UNIX® systems is described.
- ***CCTK Administrator's Manual*** – Describes setup, configuration, and administration of CCTK. The material addresses software setup, user administration, project configuration, hardware configuration, and other topics related to administration of a CCTK-based system.
- ***CCTK User's Manual*** – Includes an overview of the CCTK architecture and a complete description of how to use CCTK for operations.
- ***CCTK PCM Telemetry Interface Reference Manual*** – Describes setup, configuration, and use of the optional PCM Telemetry Interface software module.
- ***CCTK Developer's Manual*** – Describes the optional CCTK application development environment and how to use the CCTK Developer's kit to extend and customize CCTK to a particular application or operation with your own software.

1.3 Reference Documentation

The following documents complement the material presented in this manual:

- ***T-Zero™ User Manual*** – Describes the use of the T-Zero™ process control application for CCTK.
- ***RangeNet™ User Manual*** – Describes the use of the RangeNet™ range safety application for CCTK.
- ***GLG User Manual*** and <http://www.genlogic.com> – Describes the third party software tool called GLG from Genlogic. GLG provides the graphic creation and display capability for CCTK.
- ***Exceed User Manual*** – Describes the third party software called Exceed from Hummingbird Software. Exceed provides the client display for Windows-based consoles.

1.4 CCT Help Desk

The CCT Help Desk Phone Number is (321) 264-1193 (available 8:00 am to 5:00 pm, U.S. Eastern Time, Monday through Friday, excluding holidays).

CCT provides free unlimited telephone and e-mail support for 30 days after purchase as part of our standard warranty. Customers who purchase the technical support upgrade option have unlimited telephone or email support for one year after purchase or renewal. For questions or comments about CCTK or other CCT products or services please call (321) 264-1193 and ask to speak to a customer representative. For e-mail information, contact info@cctcorp.com or visit our website at <http://www.cctcorp.com>.

It is important when calling for help with a problem to preserve the project environment as much as possible to allow CCT engineers to efficiently diagnose the problem. If at all possible, leave the project and operation in its current state at the time of the problem and call the Help Desk. If it is not possible to stop and troubleshoot during an operation, please perform the following steps:

1. Completely document all of the symptoms of the problem and as much of the state and configuration of the project as possible.
2. Copy the KPATH directory to a location outside of the project environment (e.g., */tmp/projectname/date/*).
3. Note the time and date of the problem.

Additional help may be found at the CCT Technical Support web site <http://www.cctcorp.com/support>. Also, please review the RELEASE_NOTES file in the root directory of the CCTK CD for additional information and instructions.

2 CCTK THEORY OF OPERATIONS

This section describes the CCTK architecture and its theory of operation. This section should provide you with a basic understanding of the major architectural elements of CCTK and how they function together from an operator's perspective. A basic understanding of computer systems and software is presumed.

Other CCTK manuals, such as the *CCTK Administrator's Manual* and the *CCTK Developer's Manual*, are often referenced within this section for more in depth discussion of various topics. However, these documents are not required in order to gain a basic understanding of CCTK.

2.1 Theory of Operation Overview

CCTK is comprised of four major architectural segments: external interfaces, the real-time kernel, domain applications, and support utilities (see Figure 2-1). Commands and data are concurrently passed to and from the domain application segment and to and from the external interface segment. The real-time kernel provides the middleware services for quick and reliable processing and distribution of all transactions in either direction. Support utilities provide services for recording and analysis as well as project configuration management. These four segments make up a generic framework from which a broad range of control system solutions can be derived for human control and monitoring, decision support, and autonomous operation of end items. The actors in this generic framework, as discussed throughout the remainder of this document, are Users, Developers, Administrators, and End-Items.

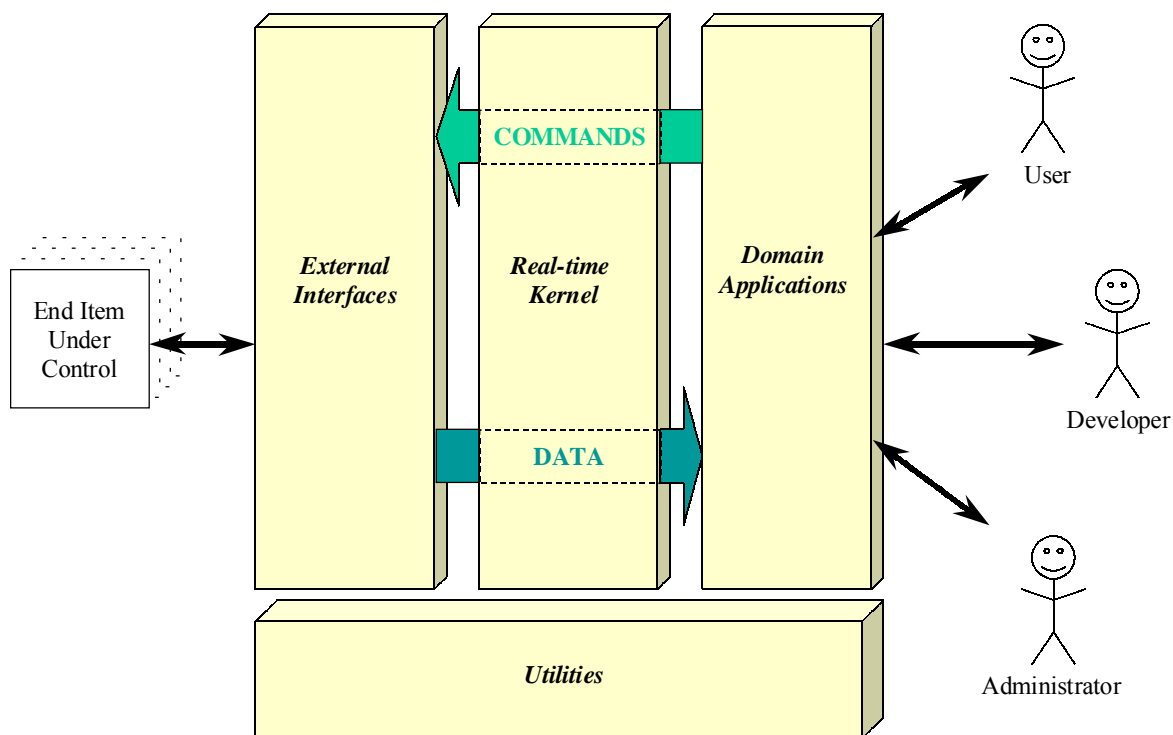


Figure 2-1. CCTK Context

The CCTK “user” is the run time operator who uses CCTK and its supporting tools to monitor and control one or more end items. A user interacts with CCTK via the CCTK graphical user interface to activate or join a real-time multi-operator activity, select various views of end-item data, initiate control actions, or obtain analysis reports. The typical CCTK operator should be familiar with basic computer operation. Power users can benefit from a basic understanding of UNIX and scripting languages such as Tcl.

The CCTK “administrator” is responsible for configuring CCTK for real-time operations activities. Most administration activities are performed off-line and separate from real-time operations. The administrator must have a good understanding of the underlying product architecture because many of the administrator’s tasks include definition of detailed configurations that, if improperly performed, can critically affect CCTK operation. The CCTK administrator should also be familiar with CCTK user operations, UNIX system administration, and have a working understanding of Extensible Markup Language (XML).¹

The CCTK “developer” creates customizations for CCTK that address specific domain application needs. The developer works off-line from real-time operations, building graphical displays and/or real-time user application programs. The developer should be familiar with the CCTK GUI builder application for graphical display creation. Developers who author custom application programs should be conversant with the CCTK application development library, CCTK simulation engine, basic UNIX programming, and the ANSI C programming language.

¹ XML is an open technology defined by the World Wide Web Consortium (W3C). see <http://www.w3.org/XML/>

The “end-item under control” can be any device, instrument, system, or group of systems that are capable of being controlled or monitored via a CCTK external interface. CCTK is configured to communicate with end-items using standard plug-in or user customized interface modules. All end-items are characterized within CCTK as a collection of related sensors and effectors (measurements and commands). CCTK becomes aware of end-item sensors and effectors when the CCTK administrator defines the end-item interface in CCTK. Because the end-item interface description is captured in a database, CCTK can change the way it communicates with the end item simply by changing the database.

2.1.1 Quality Attributes and Features

CCTK is designed to support the following major architectural quality attributes:

Table 2-1. CCTK Quality Attributes

Description	
Scalable	▪ Support embedded application to distributed multi-user client/server system
Flexible	▪ Configurable to many applications using commonly available skills ▪ Support multiple concurrent external interfaces
Extensible	▪ Customizable user interface ▪ Open multi-language programming support ▪ Command and control product line support
Reliable	▪ Integrated health management ▪ Resilient to failure
Sustainable	▪ Standards based platform independent architecture ▪ Modular functional components ▪ Receptive to integration of new technologies (Evolvable)

CCTK provides the following major functional features:

- Concurrent I/O interfaces support data acquisition, simulation, and end item commanding using commercial I/O components and industry standard protocols. Interface customization is supported with generic processing components.
- Built in applications for visualization and control such as: virtual strip charts, user configurable quick look displays, and messaging.
- User development environment for customization that includes C/C++ CCTK programming library, scripting environment based on Tcl, and GUI builders for creating dynamic graphical control and monitoring displays.
- An integrated simulation environment that supports user application development, missing element modeling, and user training scenarios.
- Master time acquisition, synchronization, distribution, and general-purpose user timer.
- Modular data processing scheme that allows processing algorithms to be dynamically chained together and extended with user custom modules.
- XML configuration databases for management of interfaces, commands, measurements, and resources in an open, extensible manner.

- Digital command, measurement, and message recording, retrieval, and playback for post mission data reduction and analysis.
- Support for plug-in applications T-Zero[™] and RangeNet[™].

2.1.2 Operations Process Flow

The CCTK operational process flow as depicted in Figure 2-2 consists of configuring CCTK (pre-project execution phase), executing a real-time operation (project execution phase), and analyzing the results of real-time operations (post-project execution phase). This thread of activities is defined as a “project.”

Prior to project execution, the CCTK administrator configures new user accounts and their associated environments, defines CCTK real-time resources and external interfaces, and populates the project configuration database with interface details, and measurement and command specifications. If the project requires customization, the CCTK developer creates user application programs and graphical displays for use during project execution. Simulations are used to verify the user application programs and graphical displays, as well as support validation of the configuration database. The simulations can also be used during project execution for missing element modeling and creating user-training scenarios.

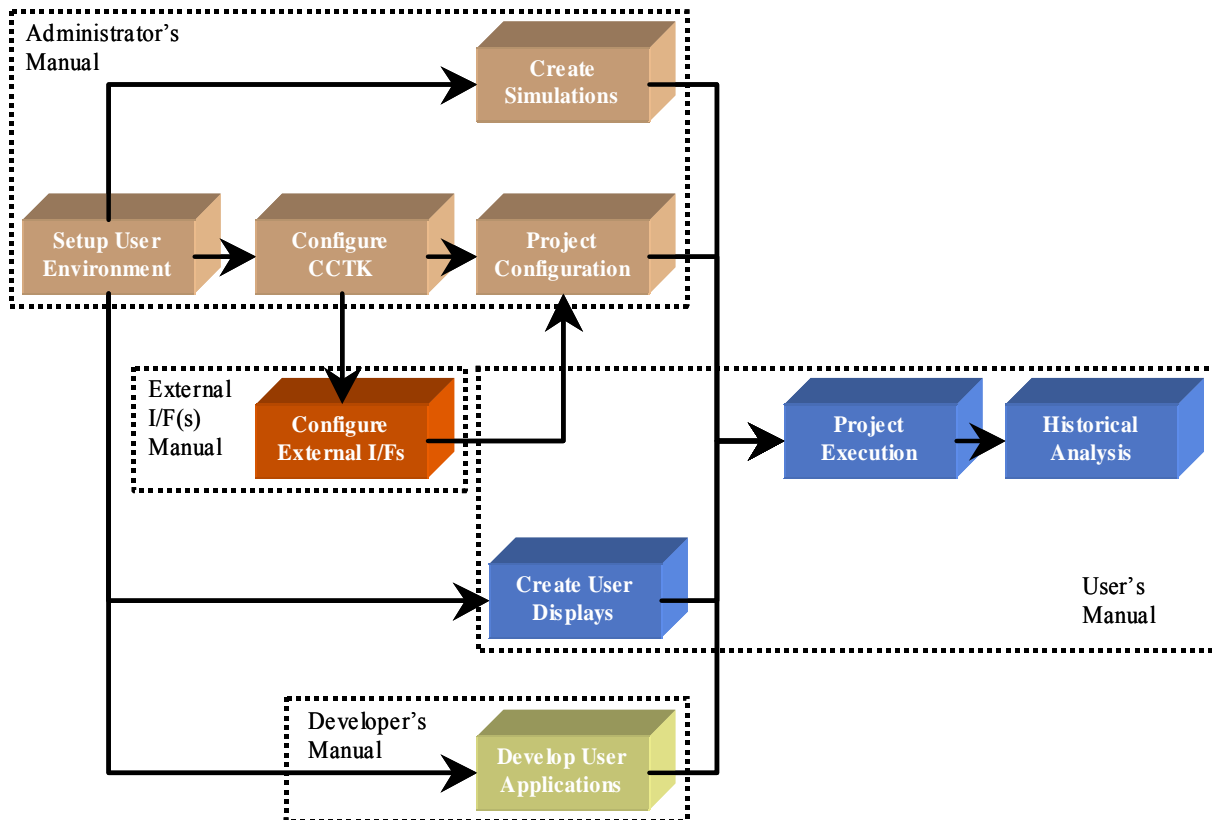


Figure 2-2. CCTK Operations Process Flow

The user executes the project created in the pre-project execution phase using the CCTK tools and customizations to control and monitor one or more end-items. During execution CCTK digitally records all real-time project information for near real-time and historical analysis.

CCTK retrieval tools are used to conduct historical analysis by producing time correlated reports and data exports that can be created and ingested by other trending and analysis tools. Batch report processing is used when analysis requirements are large and repetitive across multiple operations.

The cycle of activities is repeated for each new configuration. Since project configurations are managed in a database, the user and administrator can easily extend project execution scenarios, reuse, or copy and paste elements into new project configurations.

2.2 Physical Architecture

CCTK is supported on a broad range of physical architecture configurations. Topology options range from a stand alone laptop to a multi-user control center distributed across a hundred computer systems in many geographic locations connected together using standard networking technologies.

Stand-alone configurations shown on the left in Figure 2-3 require both the server and client software to be hosted on a single computer. Embedded configurations often do not require client software.

The basic client/server configuration has a single UNIX server and an associated client workstation connected to the server via a TCP/IP network. Most CCTK graphical interfaces utilize the X client/server protocol¹ for distributed client/server connectivity. CCTK also supports a peer-to-peer interfaces that supports specialized client/server approaches for data distribution between multiple systems.

Extended client/server configuration is similar to the basic configuration, but with the addition of a local area network containing additional client nodes.

CCTK can also support custom topology variations, as shown in Figure 2-5, such as a peer-to-peer configuration for specialized high performance configurations, and HTTP for Internet based client/server arrangements as described in Figure 2-4.

¹ The X protocol is an open technology and is defined by X.org.

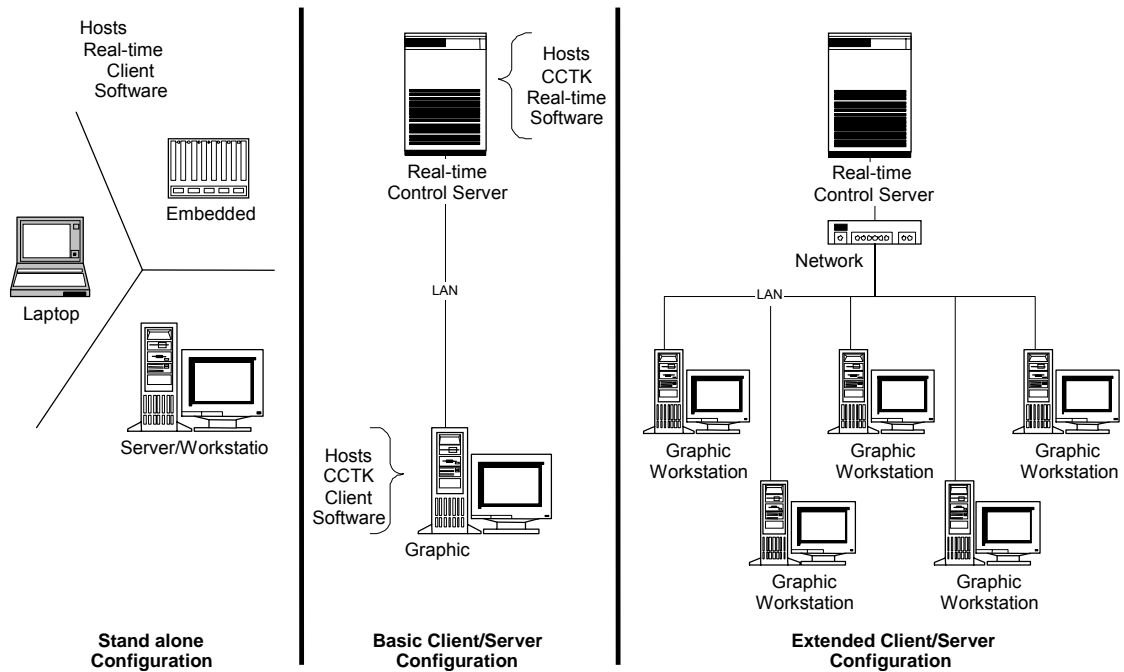


Figure 2-3. CCTK Topologies

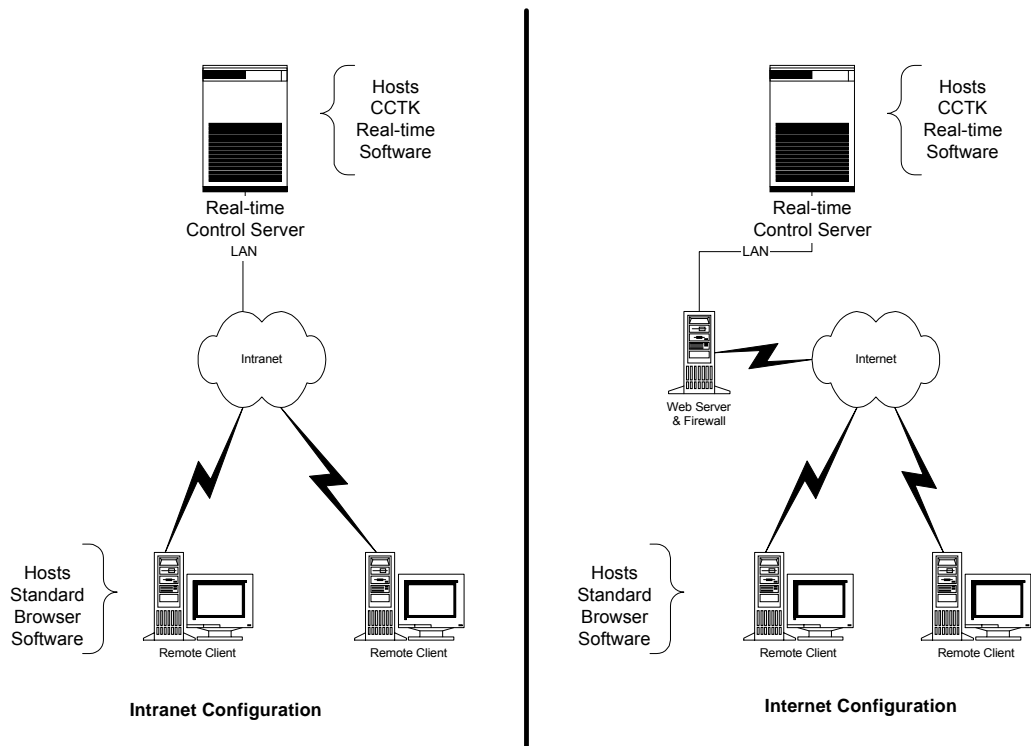


Figure 2-4. CCTK Internet Topologies

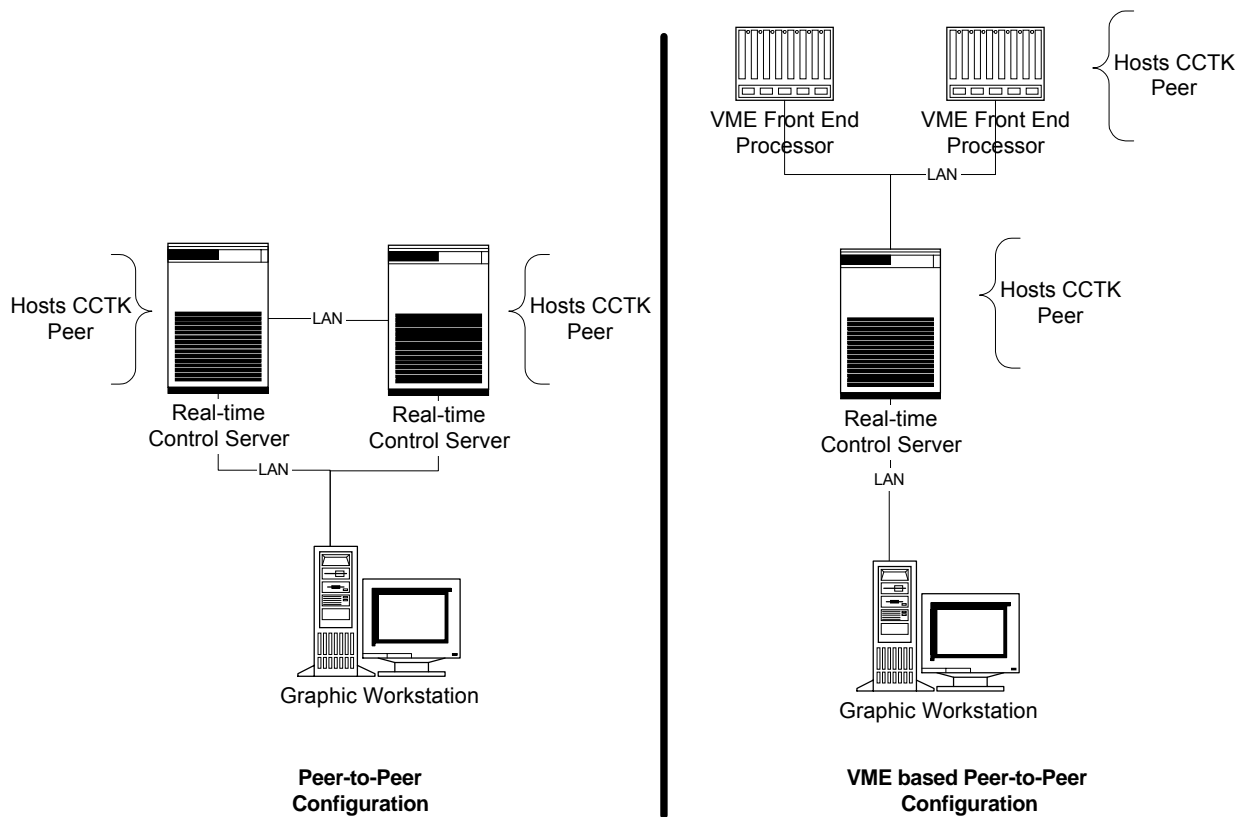


Figure 2-5. Custom Peer-to-Peer Configurations

2.2.1 Architecture Layers

CCTK employs several architecture layers as illustrated in Figure 2-6. The application service and real-time kernel service layers shield system and user applications from the external interfaces, abstracting interface details from applications, making them more sustainable and less vulnerable to technology changes. The real-time kernel services further encapsulate these

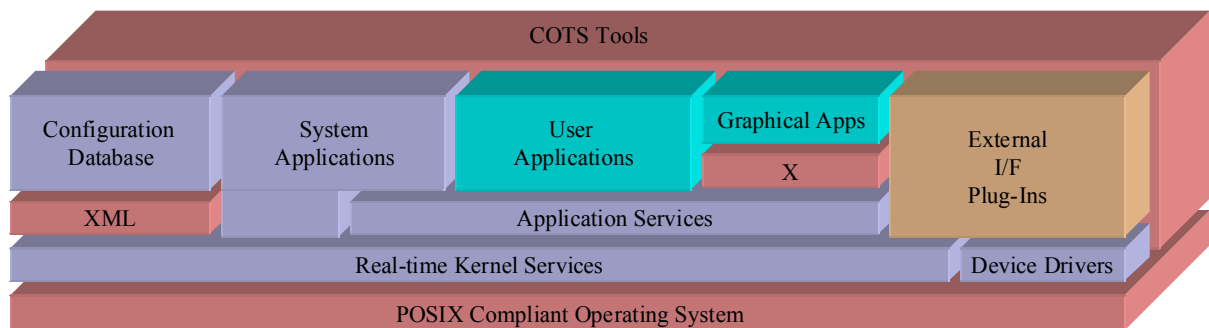


Figure 2-6. CCTK Architecture Layers

layers. Layers such as the Portable Operating System Interface (POSIX),¹ and X provide portability and degrees of platform independence. CCTK strictly adheres to the POSIX

¹ POSIX is an operating system interface standardized by ISO/IEC, IEEE and The Open Group.

standard for cross platform compatibility. Additionally, X client/server technology allows CCTK clients to run on almost all common operating systems or windowing/graphical environments.

2.2.2 Platforms

The CCTK host computer, sometimes referred to as a real-time control server (RTCS), runs on various commercially available hardware platforms that include x86 (or compatible) and RISC CPUs. Physical configurations include single board computers or parallel processing environments enabled by symmetric multi-processing (SMP) or other similar technology. Platform configurations can vary based on the application needs and performance requirements. Peripherals such as disk drives, networking, I/O buses and devices, mass storage devices, and even the presence of a console with keyboard are configured to meet operational needs. The optimum amount of computer memory is also established based on performance requirements.

CCTK requires a POSIX compliant operating system, particularly with regard to the real-time extensions defined under POSIX P1003-1b. Strict use of this standard ensures portability of CCTK software across almost any compliant platform.

The CCTK client can be hosted on the RTCS or on a workstation connected to the RTCS via a TCP/IP network. Although it is not required, the workstation is usually a graphic desktop computer running X client emulation software. This can be any UNIX class system or Windows-based PC. There are no other special configuration requirements for client workstations. However, the look and feel of the CCTK client can vary significantly based on the choice of desktop and windowing environment.

2.2.3 External Interfaces

CCTK supports external interface communications modules based on industry standard I/O bus form factors VME¹ and PCI, or any native device I/O mechanism supported by the RTCS platform (see Figure 2-7). CCTK also supports extended I/O bus protocols such as Fieldbus² for remote interface processing.

¹ VME bus (Versa Module Europa) is a flexible open-ended bus system which makes use of the Eurocard standard. It is defined by the IEEE 1014-1987 standard.

² Fieldbus is an open standard for industrial control I/O buses defined by ANSI/ISA-50.02, Part 2-1992

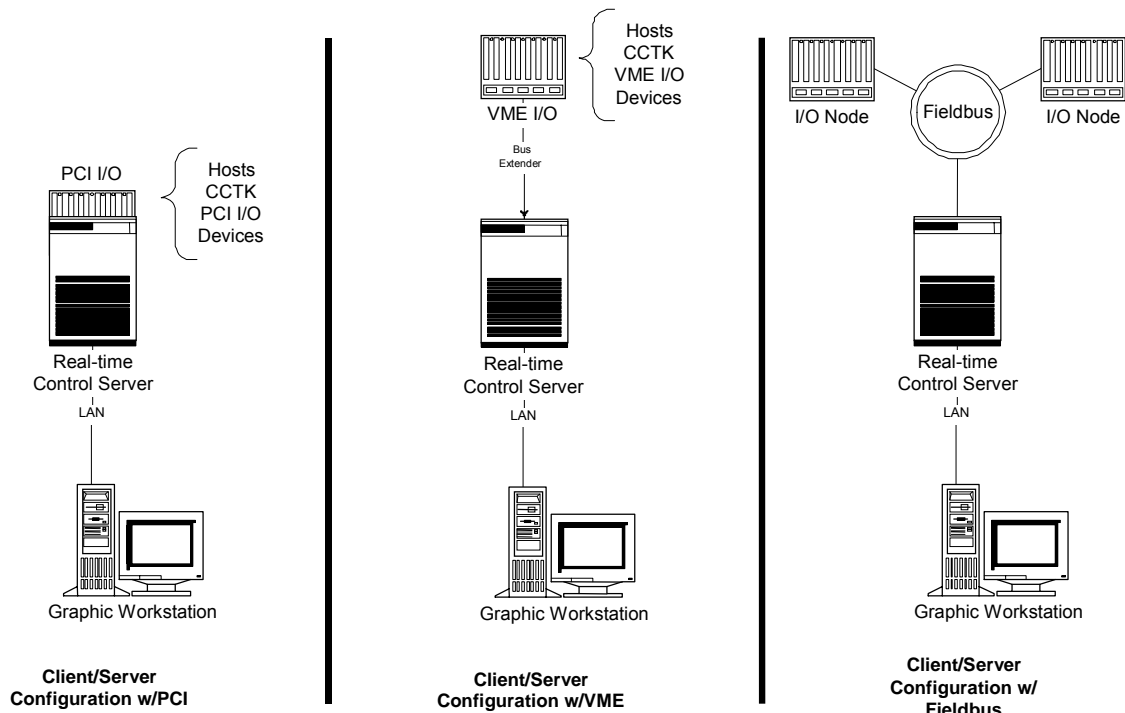


Figure 2-7. CCTK I/O Configurations

2.2.4 Networks

CCTK operates using industry standard networks and protocols for communications between clients and servers. In particular the TCP/IP based X Windows protocols are used to connect clients on a local area network (LAN) to the CCTK real-time control server. While the protocol and types of connections are standard, the following paragraphs describe some issues to consider before attaching a CCTK RTCS to a network.

CCTK clients are “thin clients.” That is, using the X Windows protocol, they present a fairly light load to the LAN that it is connected. However, too many clients, or a congested LAN may prevent data from reaching the client in a timely manner. You should consider the total network traffic that the LAN will receive and choose the physical network architecture accordingly. For example, an already congested office network running a 10 Mbps 10BASE2 Ethernet is a poor choice for connecting a CCTK server and client. A 100 Mbps switched Ethernet should be considered for all new installations.

Access to the Internet is a requirement of most businesses and research institutions in operation today. As such, the RTCS may be connected to a network with Internet access.

X windows is a relatively unsecured protocol and therefore subject to exploitation by hackers. Avoid configurations requiring client graphics workstations that connect to the RTCS over the Internet. Besides the non-deterministic delay between client and server, you are potentially opening up the server to intrusion. Given the potential criticality of a CCTK based system you should consider placing a firewall between the CCTK servers and clients, and any other LAN connected to the Internet, even though that LAN may have a firewall between it and the Internet as well.

Most firewalls do not default to allowing X window protocols through. This may require configuration by a network administrator. In sites using the optional peer-to-peer multicast protocol, system or network administrators may have to register the multicast address used with firewalls, network routers, or switches. Also, the default IP “time to live” value should be kept small to prevent multicast packets from flooding a network. Both of these situations may result in large LAN installations from being unable to receive real-time data.

The diagram shown in Figure 2-8 depicts a recommended network installation using a CCTK server, clients, and the Internet.

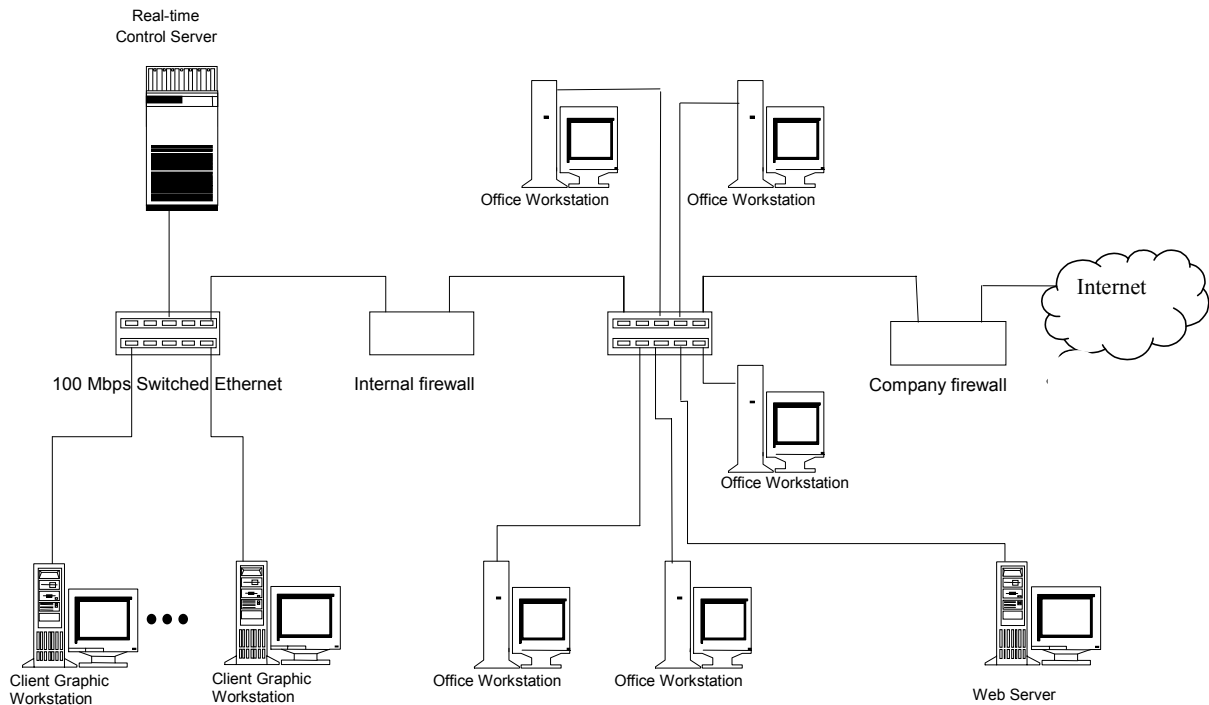


Figure 2-8. Sample Network Configuration

2.2.5 Performance Considerations

Selection of CCTK platform configuration is driven in part by the performance needs of the application. As a general rule, the more demanding the application, the more computing resources required. The optimal platform configuration provides only enough resources to meet the needs of the application (excluding growth margin). However, determination of the optimal configuration is not an exact process, and must be considered carefully in order to avoid systems deficiencies or over specification. This section discusses the performance issues that affect CCTK platform configuration choices.

2.2.5.1 Real-time Performance Background

CCTK software is designed to operate as a soft real-time middleware application; as such, it relies principally on the scheduling and process dispatching services provided by the operating system on which it runs. This approach places the burden of meeting timing constraints on the OS.

A real-time system provides specified services that meet specified timing or latency requirements. Real-time systems are generally characterized as either “hard real-time” or “soft real-time.”

- **What is hard real-time?** – A system is considered hard real-time if it must meet precise timing requirements, be they a few microseconds, a few milliseconds, or even a few seconds, depending entirely on the application. When a hard real-time system does not meet a timing requirement it must detect the error and transition to a recovery mode.
- **What is soft real-time?** – Soft real-time systems require timeliness to a degree, but do not necessarily have to meet timing requirements in order to function properly. A soft real-time system is one in which its users desire the system to respond optimally, but is not considered to have “failed” when the system doesn’t meet desired response times.

The soft real-time nature of CCTK allows a system to operate on a wide variety of hardware and software platforms since all that is required is a generic POSIX compliant operating system like UNIX.TM Special non-standard services typically associated with hard real-time systems would dramatically limit CCTK’s ability to scale and adapt to many different types and sizes of applications.

2.2.5.2 Performance Criteria

CCTK performance can be categorized based on the criteria and factors described in Table 2-2. The table identifies the major issues that affect configuration selection.

Table 2-2. CCTK Performance Criteria

Criteria	Factor	Description/Explanation
End Item(s) Complexity	Number of interfaces	The number of interfaces will drive the quantity of required I/O bus slots and may affect the type of computing platform. Laptop configurations can typically only support 1 I/O bus. Typical PC configurations can support up to 6 PCI slots. Bus expansion options or a VME form factor is required for higher interface counts.
	Types of interfaces	Support for high-speed interfaces such as telemetry can significantly affect platform selection, I/O form factor, and OS choice for a given application.
	Physical distribution of interfaces	Remote control over long distances may require extended I/O buses such as Fieldbus, or even use of CCTK Peer-to-Peer configurations.
	Measurement samples/sec	The aggregate sample rate of data from end-item interfaces affects the volume of information being digitally recorded and/or delivered to domain applications. For high sample rate applications (i.e. 50K+ s/s) it is recommended that some data modeling be performed to properly size disk controllers, and memory.
	Types of measurements and conversions	Analog data typically requires as much as 8X the amount of physical memory resources as discrete data and implies a higher data processing workload to execute FP operations. Also conversion of a ninth order polynomial will consume significantly more processing resources than a first order polynomial.
	Use of data compression	CCTK employs user tunable data compression algorithms that can dramatically reduce throughput requirements in a high performance application, and hence reduce the need for expensive processing resources. However, compression is not always an option.

Criteria	Factor	Description/Explanation
Domain Application Workload	Number of control loops	The number of control algorithms and/or computations can impact system memory resources and overall processing load.
	Application response time	Because CCTK is based on soft real-time, typical configurations cannot guarantee application response times. However, processing efficiency within CCTK make it possible to meet processing deadlines less than a few milliseconds, depending on configuration and other concurrent activities. When algorithmic performance is mission critical UNIX process locking schemes are used to optimize resource availability for execution.
	Application resource consumption	In a virtual parallel-processing environment such as UNIX, user applications are not constrained in any way. This is good and bad. It's good because it gives the application developer tremendous flexibility in constructing customized automation. It's bad because it gives the application developer the ability to overwhelm system resources if care is not taken to design efficiently. Application developers for mission critical systems should be trained in good real-time programming techniques and the theory of operation of CCTK.
Operations Complexity	Number of simultaneous users	The number of simultaneous users of CCTK will directly affect the amount of required memory for execution of a user session. The amount of memory required for each user will vary based on the types of displays used in a session, but as a general rule of thumb each simultaneous user requires 32 Mbytes of additional memory.
	Recording requirements	The speed and duration of aggregate data throughput is directly proportional to the required performance and size of recording devices. For example: A typical 1 Mbit telemetry stream can generate 1 Mbyte/sec of processed archive data, which translates to about 1 Gbyte of storage consumption every 17 minutes (assuming no compression is used).

Table 2-3 summarizes the major parameters that can be tuned to support variations in performance requirements of CCTK.

Table 2-3. CCTK Performance Configuration Parameters

Configurable Parameters	Description
Operating System	CCTK is designed to run on any POSIX 1b compliant OS, but not all operating systems are equal. While Linux is perfectly acceptable for most applications, providing cost and flexibility advantages, it is an OS that is still maturing in performance and real-time capabilities. It may be more appropriate to use a more real-time proven OS such as IRIX™ from Silicon Graphics, which has been a leader in multiprocessing high performance systems for many years.
Processing Capacity	CCTK provides no special features beyond what is typically provided by the OS for processor management. CCTK can run on a single processor computer or a multiprocessor computer supported by SMP or equivalent. A multiprocessor approach is often taken when larger margins are needed to support critical processing or future growth. Note: Many multiprocessor systems can be field expanded with more processors without affecting systems architecture.
Physical distribution of processing	Since the world is distributed, so is the system that controls it. Integration with legacy systems, and access to remote end items may require distributed processing topologies. CCTK can be adapted to central processing or distributed.

Configurable Parameters	Description
Amount of memory	Since CCTK relies on UNIX virtual memory management, it's possible to run CCTK with a very small memory footprint, particularly for embedded applications. However, once memory requirements exceed available resources, process swapping occurs, significantly slowing down system performance. Swapping is very undesirable for mission critical applications, or applications that have timing deadlines. The minimum memory configuration for a single user system is 128 Mbytes. As the number of simultaneous users or other performance drivers increase, so should the memory configuration.
Size of archive devices	The maximum size of a single contiguous archive is limited only by the maximum file size supported by the host OS. The size of archive drives is dependent primarily on desired number of project archives that need to be available for data reduction and analysis. If the max archive file size is 2 Gbytes, multiply the number of desired historical recordings to arrive at an approximate capacity. Permanent storage media such as DAT or optical disk can also be used to manage historical archives if disk space is limited. Additionally, it is recommended that archive drives be separate physical devices from systems drives (OS and user space) for mission critical systems.
Speed of archive devices	Aggregate systems throughput is the primary driver for selection of high-speed archive drives and controllers. SCSI and/or RAID configurations are recommended for medium to high performance mission critical applications.
Network selection	CCTK client networks rely on standard TCP communications. Processing load management is typical of any Local Area Network. Multiple networks can be employed as required to increase bandwidth.

2.2.5.3 Example Performance Profiles

Table 2-4 describes a range of example performance profiles that should help with characterizing and sizing systems. The table describes a typical system type, quantity of processors, amount of memory, number of users, and expected throughput bandwidth.

Table 2-4. Example Performance Profiles

System Type	Processors (Qty)	Memory (Mbytes)	Number of Users	Throughput Performance (Samples/Second)
Minimum Embedded (SBC)	1	32	0	1K – 10K
Medium/Maximum Embedded	2	256+	0	10K – 50K
Minimum Laptop	1	64	1	500 – 2K
Minimum Single User Workstation	1	128	1	5K – 20K
Minimum Single User Server	1	128	1 – 3	10K – 25K
Medium Server Multi-user	2	256+	2 – 10	20K – 100K
High Performance Server Multi-user	4 – 16	512+	10 – 50	100K – 500K+

The numbers in the table are based on typical COTS processing equipment available at the time this document was written.

2.3 Software Architecture

CCTK software is comprised of four major architectural segments: external interfaces, real-time kernel, domain applications, and support utilities. Commands and data are concurrently passed to and from the domain application segment and to and from the external interface segment. The real-time kernel provides the middleware services for quick and reliable processing and distribution of all transactions in either direction. Support utilities provide services for recording and analysis, as well as project configuration management. Figure 2-9 is a high level view of the CCTK system architecture that shows the major software components within each segment.

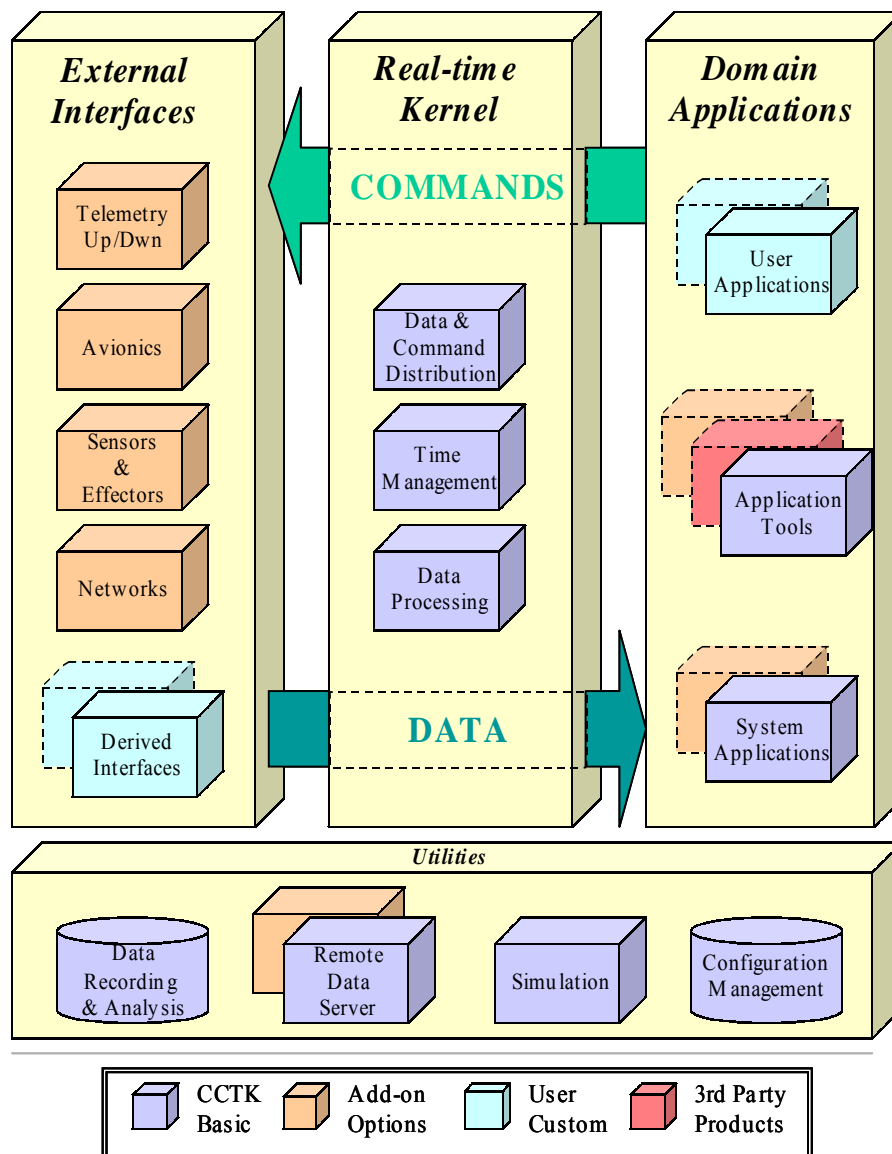


Figure 2-9. CCTK Logical Architecture

This section discusses each of the segments and their major software components. Figure 2-9 uses a legend to differentiate categories of software that makes up the CCTK configuration:

- CCTK Basic - Software included as part of the standard software CCTK configuration
- Add-on Options - software purchased separately that plug into CCTK to provide specific capabilities
- User Custom - indicates areas for customization by the user
- Third-Party Products - commercially available extensions and upgrades for CCTK

2.3.1 Real-time Kernel Segment

The real-time kernel segment provides the core middleware foundation on which any CCTK configuration is deployed. This segment provides data and command distribution, data processing, and time management elements. These core elements are configurable to support a variety of operational scenarios.

2.3.1.1 Data and Command Distribution

CCTK employs two approaches to real-time data and command distribution: message based interprocess communications called channels, and a real-time database or current value table (CVT) query. Both approaches are relatively transparent to the typical CCTK user when interacting with CCTK through a graphic display. However, the underlying schemes for data access are very different, and may impact how data is interpreted and used.

The primary purpose of the channel mechanism is to provide a very high-speed point to point and broadcast message-passing scheme between real-time CCTK processes. The channel resources are configured as part of the project configuration process performed by the system administrator. Data is passed through channels in discrete groups called “packets.” Packets can contain:

- Processed and linked data
- Events (exceptions)
- Command and command response packets
- System messages

The channel communication mechanism is also available to CCTK applications. See the *CCTK Developer's Manual* for details on using channels in user applications.

The CCTK real-time tables provide fast access to the current measurement values and associated attributes of any data or command defined within an active project. Most user applications utilize this mechanism for cyclic update of data displays. The real-time tables are a very efficient means of disseminating real-time information, although they should not be used when access to every data sample is required.

2.3.1.1.1 Real-time Tables

CCTK maintains two separate databases of information to meet the real-time performance requirements associated with high-speed data acquisition: an off-line (XML) configuration database and a memory resident database (real-time tables).

The CCTK off-line or pre-execution configuration database is stored in one or more XML files. These XML files contain all of the information necessary to configure the system for each project. These files may be modified at any time, but the information stored within them is ONLY read by the CCTK system at initialization. Therefore, if a change is made to the XML configuration files while a project is running, the project must be restarted before the configuration change will take affect.

The CCTK on-line (real-time) or execution database uses the configuration information stored within the XML files to generate the run-time information necessary for operation of the system. This information is stored in a series of shared memory segments called real-time tables. When a project initializes, it creates a set of real-time tables, as specified by the configuration database. Any process attached to the CCTK project can access the tables using the CCTK application programmer's interface (API). The CCTK API provides a consistent, safe manner to access the information stored in the tables. If, during project execution, a change is made to the configuration information stored in the real-time tables, that information IS NOT transferred back to the XML configuration files. It is up to the project administrator to ensure that changes made to the configuration during real-time operations that need to be permanent changes are propagated to the XML configuration files.

Each CCTK project has more than one real-time table. Multiple real-time tables are used to organize the data into logical sections. Multiple tables also protect data by isolating inadvertent or damaging access to a single table. All CCTK projects must define the tables listed in Table 2-5.

Table 2-5. Required CCTK tables

Name
Notice Descriptor Table (ndt)
Measurement Descriptor Table (mdt)
Command Descriptor Table (cdt)
Link Descriptor Table (ldt)
Port Descriptor Table (pdt)
Bus Descriptor Table (bdt)

2.3.1.1.2 Descriptors

Each CCTK table stores data as a series of descriptors. A descriptor is a logical grouping of related information that defines an item. For example, within CCTK there are analog measurements. Each analog measurement has a set of attributes or properties including name, description, units, conversion information, current value, etc. Each analog measurement is defined in the system as an analog measurement descriptor where all of its attributes are stored. Each descriptor is stored in the appropriate real-time table; in the case of an analog descriptor, they are stored in the measurement descriptor table.

Each descriptor has several attributes that are common to all descriptors. These common attributes include a name, a description, and a system identifier. The name, which is referred to as the functional designator (FD), is a unique tag used to request information about that descriptor. There is a one to one mapping between an FD and a descriptor. FD's must be unique within a project. Most CCTK applications reference FD's. For example, when a user selects the measurements to view in the Measurement Monitor application, a list of

measurement FD's is presented. When a user selects notices to filter in the system message GUI, a list of notice FD's is presented.

Another attribute that is common to all descriptors is the internal system assigned system identifier (SID). A SID is a dynamically assigned 32-bit number that identifies a descriptor. Again, there is a one to one mapping between descriptors and SID's, and therefore a one to one mapping between SID's and FD's. SID's are guaranteed to be unique for each project. A typical user will not have access to the SID's, since SID's are mainly used internally to improve performance. Some lower level API calls require a SID as an argument. Functions for converting from a FD to a SID and vice versa are provided in the CCTK API.

2.3.1.1.3 Channels

CCTK also uses an inter-process communication mechanism called channels. Channels are similar to message queues, but are implemented using shared memory and spin locks. Since channels use shared memory, and generally don't use system calls, they are an efficient and fast method for passing data between UNIX processes. Channels are used for most communications between CCTK processes. Each channel has one and only one client who reads data from the channel. Each channel can have multiple servers who write data to the channel. Thus, there is a many-to-one relationship between channel writers and channel readers as shown in Figure 2-10.

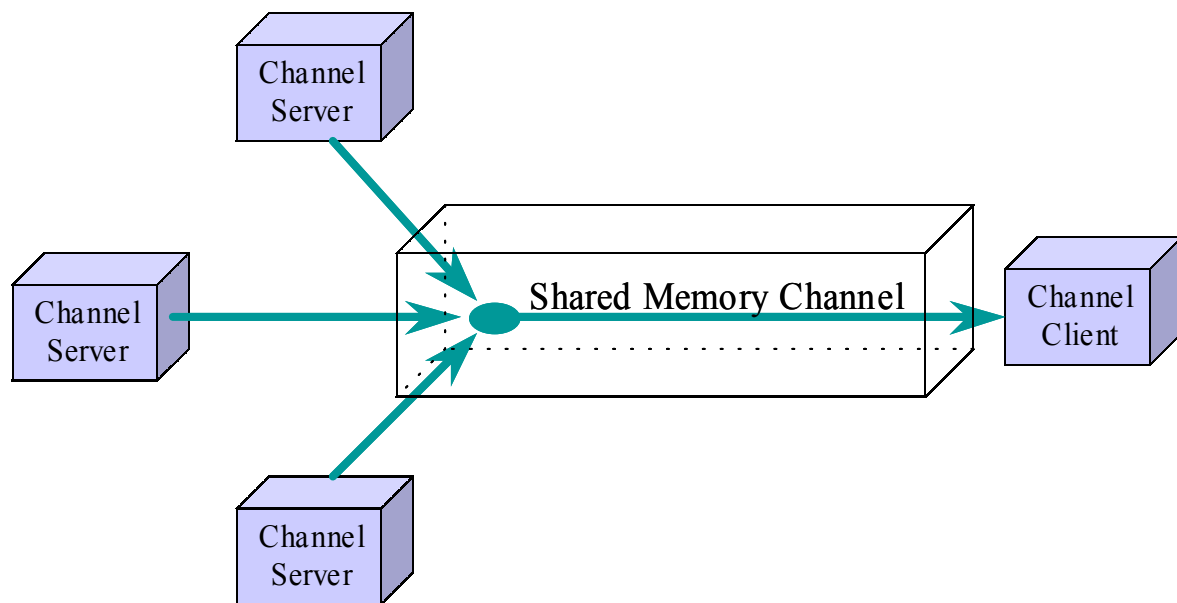


Figure 2-10. General Channel Architecture

Channels are configured on a per project basis. Each project can control the number of channels configured as well as the size of each individual channel. Depending on the project requirements, it may be necessary to size channels differently. Details on configuring channels are provided in the *CCTK Administrator's Manual*.

For proper CCTK operation, five standard channels must be defined as described in the *CCTK Administrator's Manual*. These channels are used by the system to pass data to CCTK system processes. In addition, the user may define named channels. These named channels are typically used to link interface processes to the system processes. The user may also define

unnamed channels. Processes that require a channel for internal communication with the system request unnamed channels. Command responses typically flow over unnamed channels.

2.3.1.1.4 Status

CCTK continuously calculates the status of a running project and reports that status to the user. For most users, status is always present in the lower right corner of CCTK Client.

Status is composed of two elements: state and health. State indicates the operational state of the project. State is an indicator of whether or not processes may attach to the project to perform operations. CCTK states are described in Table 2-6 below.

Table 2-6. CCTK State Definition Table

State	Description
DOWN	The open project is not running (CCTK is inactive). This state is displayed in yellow.
STARTING_UP	The open project is transitioning to the UP state. When the project is in the STARTING_UP state, it is not yet ready to service requests from the user. This state is displayed in yellow.
UP	The open project has been successfully started and it is ready to service user requests. This state is displayed in green.
SHUTTING_DOWN	The open project is transitioning out of the UP state. This state is displayed in yellow.
STARTUP_FAILED	The open project was not able to startup properly. This state is displayed in red.
FORCED_DOWN	The open (previously active) project was forced to stop. This typically indicates that a serious problem occurred during startup/shutdown and the system would not respond to a normal shutdown. This state is displayed in red.

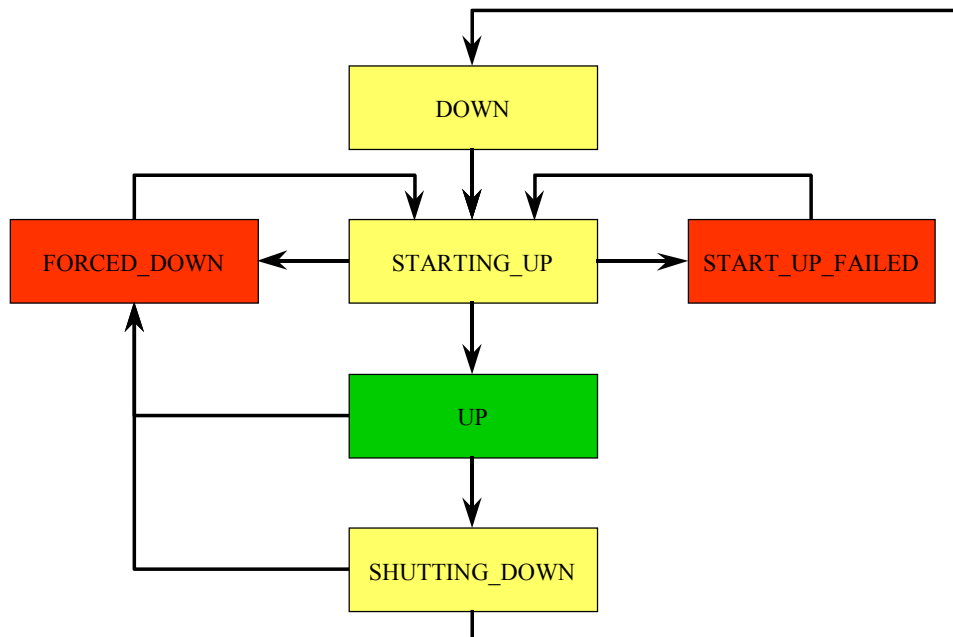


Figure 2-11. CCTK State Diagram

The user primarily controls the project state. The user indicates that a state transition is desired, typically by starting and stopping a project from within CCTK Client. The allowable state transitions are illustrated in Figure 2-11.

Health indicates whether or not a running CCTK project is operating as expected or operating with problems. The valid health states are described in Table 2-7.

Table 2-7. Health States

Health State	Description
CRITICAL	One or more serious problems have occurred with the project that will affect its ability to process data and perform operations.
CAUTIONARY	One or more minor problems have occurred with the project. Operations may be degraded, but the project should still be able to perform its primary operations.
HEALTHY	Everything is nominal.

Health is only calculated when a project is in the UP state. Health is invalid in any other state. System health is a composite of the health of all CCTK processes. Each process has the ability to report its health back to the system as one of the above items. If any one process reports back CAUTIONARY, then the health of the system becomes CAUTIONARY. If any one process reports back CRITICAL, then the health of the system becomes CRITICAL.

2.3.1.2 Data Processing

Data processing within the real-time kernel represents a uniform configurable scheme for applying various algorithms to measurement and command data. In the simplest sense, data processing can be thought of as a set of algorithms that are applied in sequence on a piece of data. Each algorithm has the capability of altering the data. A simple example of this would be a series of bytes that need to be endian swapped and then processed via a simple polynomial function as illustrated in Figure 2-12. More complex variations on this concept may include selection of alternate algorithm sequences based on in-line algorithm results, or alternate algorithms based on, and including, the derived value of other data.

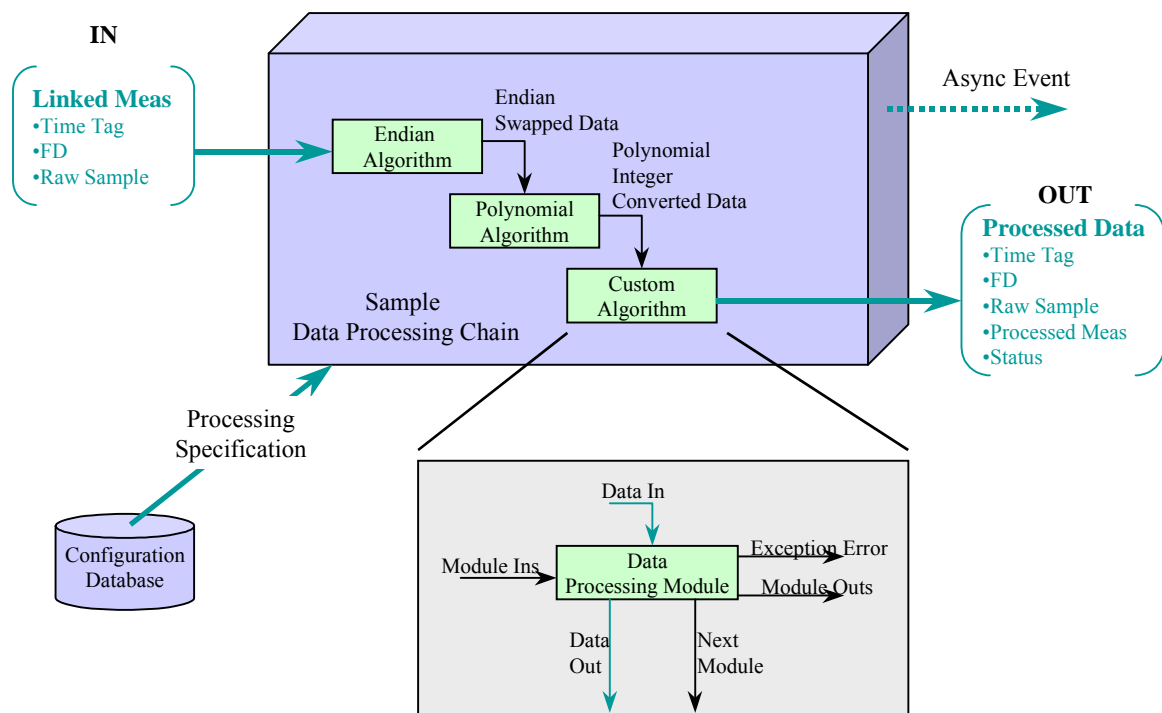


Figure 2-12. Data Processing Example

CCTK provides standard processing algorithms for performing data conversion, filtering, and evaluating data for state conditions as discussed in Table 2-8. Provisions are also made for user customization with the optional development environment (see *CCTK Developer's Manual*). Once processed, data is distributed for user application access, display, and archival. The configuration database contains the specification for measurement/command-processing algorithms that are applied in real-time (see *CCTK Administrator's Manual*).

Table 2-8. Standard Data Processing Algorithms

Processing	Description
Conversion	The raw value is converted to engineering units using a prescribed algorithm contained in the descriptor. The purpose of data conversion is to do linearization and scaling but not to convert the value of one data type into another. Data conversion is only applicable to analog type data.
Compression (filtering)	The current sample is compared against the previous sample to determine significant change. The significant change criteria can be based on the raw value, processed value, or a percentage of the processed value.
Stale	The new raw value is compared against the previous raw value. If the raw data remains unchanged for a specified number of samples, then the new sample is considered stale. The stale data bit is set in the measurement's status.
Range Validity	The measurement is verified to be within the proper range. The upper and lower range values are defined in the descriptor. They can be specified in terms of the raw or processed value. The invalid bit is set in the measurement's status if it is not in the proper range, and other exception checks and data conversions are not performed.
Max Change	The delta between the current and previous values is compared against a delta value specified in the descriptor. This value can be specified in terms of the raw, processed value or a percentage of the processed value. If the specified delta is exceeded, the appropriate bit is set in the measurement's status.

Processing	Description
Exception Limits	A measurement can have up to eight limit conditions defined. For each limit defined, the processed value is compared to the specified limit and its logic condition. An exception bit is set in the measurement's status if the logic condition is met.

Any change in the status bits associated with stale data check, valid range check, max change check and the limit condition checks can result in an exception message. The occurrence of an exception limit, range validity, max change, or stale status change can be associated with an asynchronous event notification. The notification is a message delivered from data processing to a registered application over a CCTK channel. The event can be used to initiate a reactive control sequence or graphic display dynamic. For example, discrete data inputs such as an instrument, break wire, or a status bit defined in a downlink telemetry stream can trigger an exception when it changes state (On/Off). The exception triggers an application response to save the project, initiate an event handler, and notify the system user. This event processing allows applications to idle until something important happens, saving valuable system resources. As with other processing specifications, event notices are defined in the CDB. Exceptions are also time-stamped and recorded for later analysis.

Data can be represented within CCTK as one of the following data types:

- Discrete
- Signed Integer
- Unsigned Integer
- IEEE 757 Floating Point
- Byte Array

2.3.1.3 Time Management

All data, commands, events, and messages are time stamped to a one-microsecond resolution to provide real-time or historical time correlation. System time, sometimes referred to as “universal time,” can be synchronized to an external master timing unit or be autonomously generated internally.

When synchronizing to external timing references such as GPS or IRIG-B time, CCTK acquires the reference source via a plug-in external interface device. CCTK periodically samples, normalizes, and then distributes the timing signal as processed data for display and application processing. Time services can detect time source failure or time shifts and assume time generation function internally. A user can also manually select an external reference source or internal generation.

In addition to system time, CCTK supports general-purpose timers such as countdown time or mission elapsed times that synchronize to system time. A time control graphic application provides control options to set, start, stop, and configure automated time command events during real-time operations.

CCTK also supports the Network Time Protocol¹ (NTP) standard for synchronization in distributed configurations for servers and clients.

¹ The Network Time Protocol (NTP) Version 4 specified in RFC-1305 [MIL92] is widely used to synchronize computer clocks in the global Internet.

2.3.2 External Interface Segment

The external interface segment hosts the hardware and software necessary to communicate with end-item components or systems. Common external interfaces include telemetry, avionics, time, sensor and effectors, supervisory control and data acquisition (SCADA), and network interfaces. CCT offers a number of different configuration options for external interfaces including interface simulators. In addition, the segment can be field customized to support application-unique requirements using the CCTK development option.

External interfaces are generally comprised of an I/O device, a device driver, and an interface processing software utility. The software utility handles the external to internal communication protocol transition and provides a service layer for integration with CCTK system control, health functions, and end item commanding. A graphical user interface is included for some interface types. Like all of CCTK, interfaces are configurable using the configuration database.

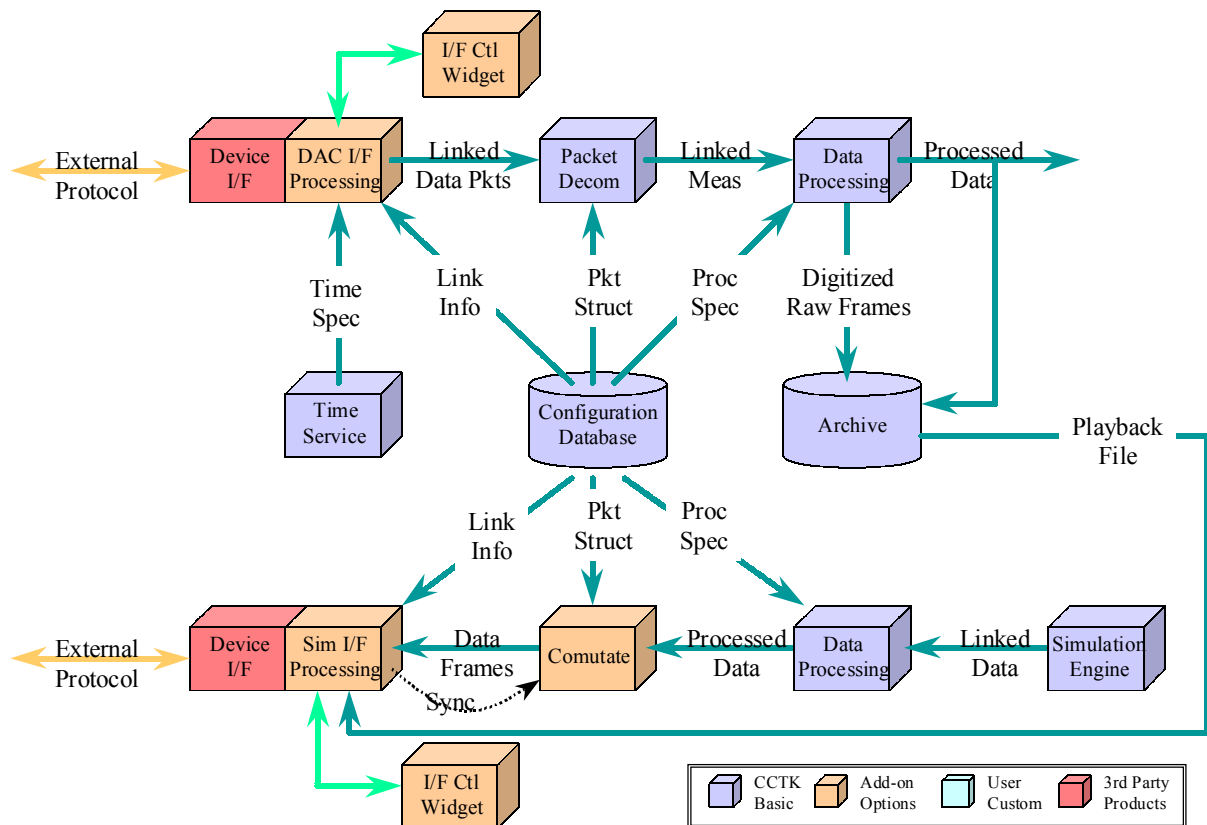


Figure 2-13. External Interface Architecture

Figure 2-13 illustrates a generic model of CCTK external interfaces. External protocols are handled by device interfaces that include both I/O boards and device drivers. Interface processing software isolates the device specifics from the decommutation/commutation processing and data processing services. These services are device independent because the interface processing software communicates via a normalized linked data packet. For data acquisition, packet decom provides a generic mechanism for parsing linked data packets into raw measurement samples that can be processed and archived.

This process works similarly when applied to simulation and end item commanding. Data or commands must be processed in reverse into raw form and commuted into frames appropriate for translation by the simulation external device. The nuances of the device, structure of frames, and algorithms for translation to and from raw and processed data forms are managed via the configuration database.

All external interfaces are unique and can introduce subtle variations in this generic model. However, CCTK is designed for extensibility, applying abstraction techniques to minimize the impact an interface has on the generic nature of the CCTK architecture and user services. CCTK will generally function the same regardless of interface type.

CCTK supports PCI and VME form factor I/O products or most devices integrated through standard operating systems services such as network and serial interfaces.

2.3.2.1 Interface Processing

CCTK is adaptable to a large number of applications largely because of the ease of integration of external interfaces. Because interface-processing structures are defined in the configuration database, it is possible to define and allocate many interface-processing configurations without configuration or application-specific software. The configuration database contains structural and hardware addressing information that defines the external source and/or destination of commands and data.

The interface software utility manages interface protocol transitions. Data is acquired, digitized, ordered into individual measurement samples, time stamped, and assigned a parameter identification into a normalized form that can be propagated for further processing or digital recording. For example, telemetry is received as a raw binary stream of ones and zeros that is accumulated by the data acquisition hardware into a digital buffer as frames or packets of information that can be taken apart into constituent measurements.

This process works similarly in the opposite direction for commands or interface simulation. CCTK can commute data/commands into defined frame structures or messages that are passed to an I/O buffer and transmitted in its base communications format.

2.3.3 Domain Application Segment

The domain application segment hosts the CCTK system applications, user applications, and application development tools. The foundation of the segment is a comprehensive “Runtime Application Service Library” for measurement processing, commanding, messaging, inter-process communications and health and status as described in Figure 2-14.

The fundamental function of this segment is to abstract the external interface protocols and other real-time services from domain applications. This allows applications to work independently of external interface protocol details. Interface configuration can change from project to project, but since all CCTK applications use a standard service layer they can remain unchanged, saving time and resources while still allowing flexibility. Without this abstraction, user application programs would have to be modified or re-written every time an interface format changed or a new device technology is introduced. The net result is that underlying technology can adapt and evolve over the application lifecycle without adversely affecting the investment in system customization.

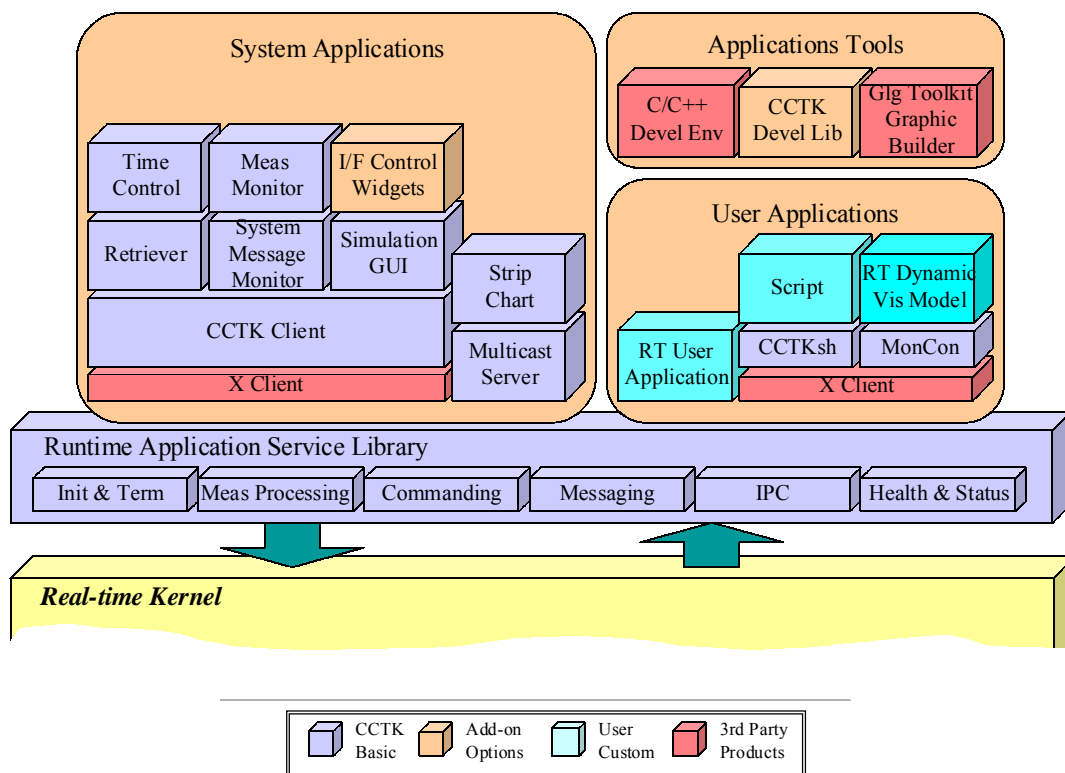


Figure 2-14. Domain Application Architecture

2.3.3.1 CCTK System Applications

CCTK provides an integrated suite of general-purpose applications for real-time operations. These system applications include:

- CCTK Client - The primary application for interacting with CCTK. It is the parent user interface for launching all other system applications.
- System Messages - A configurable message bulletin board for viewing and categorizing asynchronous messages within CCTK
- Time Control - A graphical control panel for controlling countdown time or project elapsed time.
- Real-time Measurement Monitor - A tabular graphical display for viewing real-time measurement data.
- Simulator - A graphical user interface for building basic simulation scripts and models.
- Retriever - A general purpose tool for analyzing and reporting on recorded information in near real-time or from historical archives.
- CCTKsh – A Tcl-based scripting environment and command line interpreter integrated with CCTK.
- Strip Chart - An application for viewing real-time data in a time-based graphical X-Y plot.

All of these applications are described in more detail in Section 3 of this document.

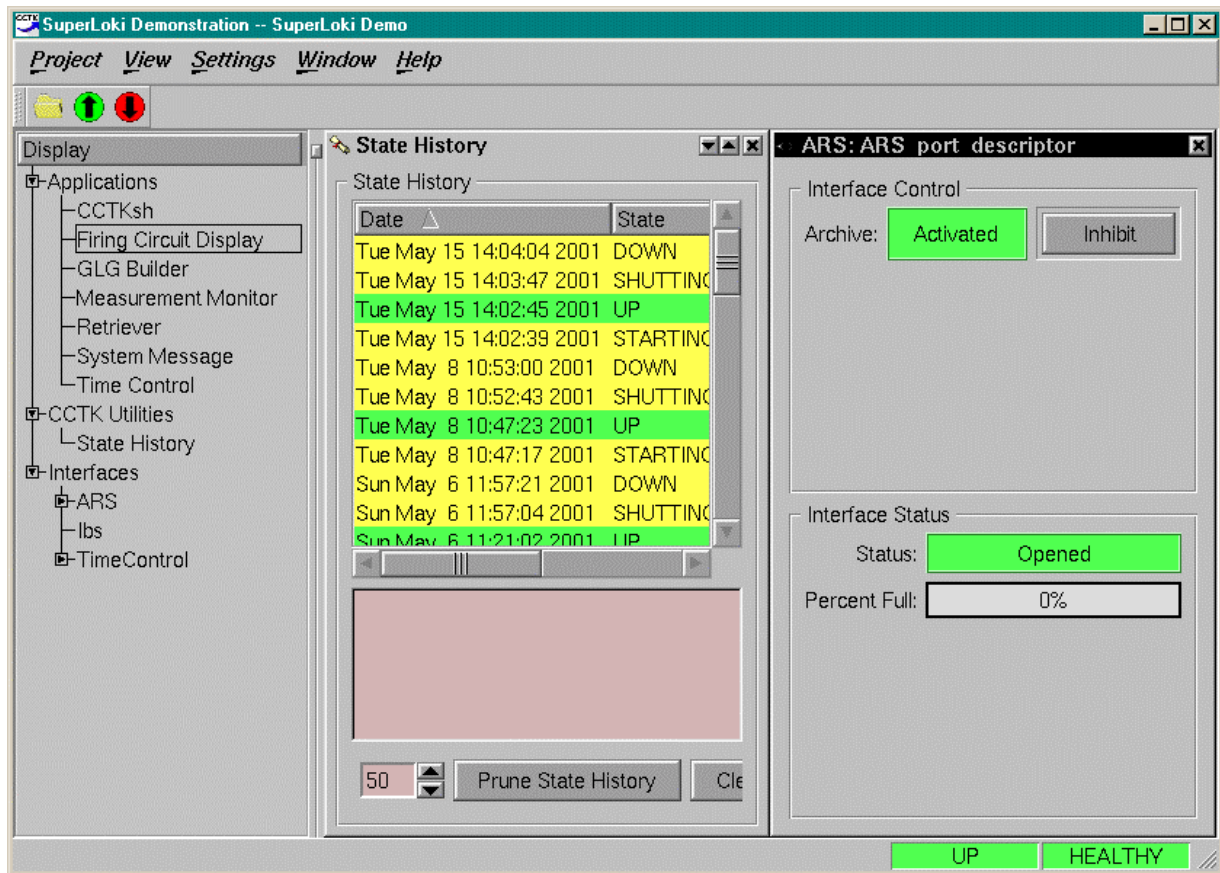


Figure 2-15. CCTK Client Graphic User Interface

User access and permission controls are defined prior to project execution and invoked via user logon authentication. The CCTK software package includes a graphical user interface constructed in a main menu format to give the user a single menu for access to all system functions, including system control and monitoring, system displays, application displays, data retrievals, and other functions.

Any application is accessible by any user/client in the system. All screens are pre-configured and selectable from the system desktop menu. Any user may create their own custom displays and bind them to real-time data within the system using the integrated GUI builder discussed later in this section.

Because all real-time data is ingested and processed by the data processing engine, any client connected to the RTCS can access any data defined in the configuration database.

2.3.3.2 User Applications

The user application environment provides a C/C++ API for data and event access, command issuance, application interaction, and creation of derived data. Control applications can be created independently of the underlying interface processing and communication protocols. Further, the underlying UNIX operating system provides services for parallel processing, file system access, and inter-process communications. The number of possible applications is limited only by the available processing resources. Applications can be developed off-line or in parallel with on-going run-time operations.

Applications can take the form of the classic finite state machine like the one shown in Listing 2-1. This example could apply to a control loop for filling a fluid tank, maintaining temperature and humidity levels in an environmental chamber, or positioning a robot arm end effector.

The CCTK event processing capability simplifies real-time control application development and minimizes the resource impact. An operator or application can register for asynchronous data-driven events (sometimes referred as exceptions). Specification of event conditions tells CCTK data processing software to monitor measurement data and notify an application when the event occurs. Event conditions are typically one or more possible values associated with a sensor that when reached can trigger a control action. Event conditions can be specified off-line in the configuration database. List 2-2 shows how the simple finite state machine (FSM) example changes when using events.

Listing 2-1: Basic Real-time Control Application

```
while (alive)
{
    get_measurement_data(); // read one or more sensors
    compute_control(); // calculate control parameters based on incoming data
    control_device(); // initiate control activity
    // read next group of sensors...calculate...control
}
```

Listing 2-2: Basic Real-time Control Application Using CCTK Event

```
sleep (until_event)
{
    get_measurement_data(); // read one or more sensors
    compute_control(); // calculate control parameters based on incoming data
    control_device(); // initiate control activity
    // read next group of sensors...calculate...control
}
```

Real-time applications are not always created for control purposes. Other types of applications include analytical computations, customized filters and data fusion, complex user interaction, and real-time modeling and simulation.

The CCTK simulation engine enhances application development by allowing an application developer to test a new application without the risky prospect of connecting to an end item device. This capability is discussed more in the *CCTK Administrator's Manual*.

CCTK applications can take the form of add-on products such as T-ZeroTM and RangeNetTM which are available from CCT for use in specific application domains. These products use the same services that are available to all CCTK application developers.

Because most of the CCTK application services are accessible via a linkable API, other third party products such as MatrixXTM, Control ShellTM, and LabViewTM can also be used to further extend the CCTK application development environment.

Details for use of the CCTK application development environment are included in the *CCTK Developer's Manual*.

2.3.3.2.1 Real-time Dynamic Graphical Displays

CCTK provides customizable graphical visualization capability for user interaction. Application displays are dynamic interactive graphical interfaces for end-item control and monitoring. The user created displays are built with a GUI builder using GLG Toolkit™ or other data visualization packages such as DataViews™ or SL-GMS.™ Some of the general capabilities of the application displays include:

- Real-time schematic model views of systems and subsystems for logical and intuitive operations.
- Selective integration of manual controls for individual device control and monitoring, and capability to initiate automated sequences for closed loop hands-off control, providing operations personnel optimum flexibility in operations support.
- On screen capability for information hiding and drill down provide for tuning of display details to fit the operational need.
- Graphics, color, and digital dynamics are used for instrumentation readout, prompts, command responses, event/state and limit violation indicators, and health and status conditions.

In many cases it is possible to meet all of a particular application's needs without writing any application software simply by creating a monitor and control display using the GUI builder.

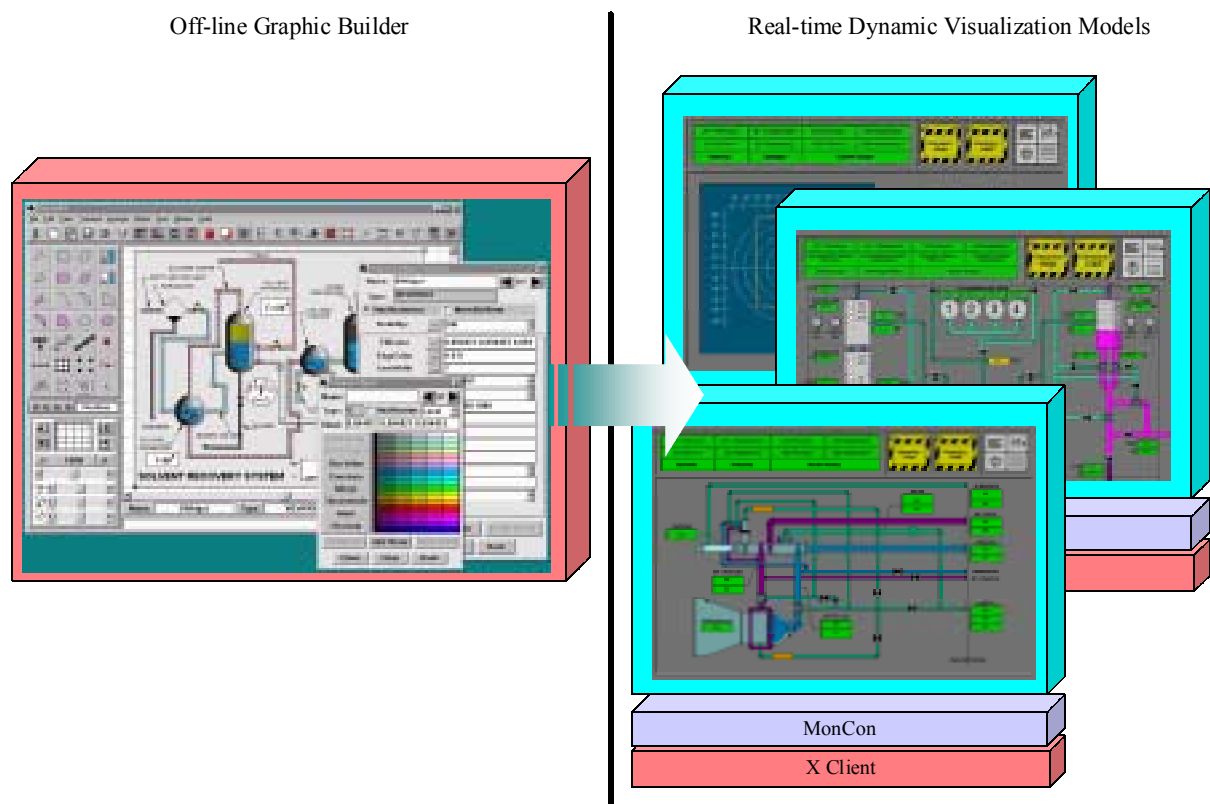


Figure 2-16. User Display Creation

2.3.3.2.2 GUI Builder

User display creation and editing is provided via the CCTK GUI Builder, which is based on GLG Toolkit.TM GLG displays are created and executed on the RTCS and displayed remotely on any workstation via X Windows Server emulation software such as ExceedTM from Hummingbird Software.

The GLG Toolkit provides an interactive environment for creating dynamic 2D and 3D graphics in a point-and-click editor. It also provides a resource-based API for supplying dynamic data and integrating graphics into the CCTK application environment.

The GUI Builder consists of the following major components:

- GLG Graphics Builder - The GLG Graphics Builder is an interactive 2D and 3D graphical editor with a point and click interface. It is used to create custom dynamic visualization components for native and web based graphics.
- Real-time Dynamic Displays Engine - A library of classes and functions for embedding dynamic graphics into an application (MonCon). It provides functionality for connecting graphics to the application data.

There are many additional components available for GLG. For more information go to www.genlogic.com.

2.3.4 Support Utilities Segment

The CCTK Support Utilities Segment is comprised of a suite of tools used to manage project configurations, record system transactions, retrieve and analyze transactions, serve remote clients and the World Wide Web, and perform simulation activities.

2.3.4.1 Configuration Management

CCTK is a generic tool that can be configured and reconfigured to perform a variety of control and monitoring tasks. Information describing a particular configuration is captured in a persistent repository called the configuration database. Each instance of a configuration is identified as a “project.” The project is given a unique name such as “Tank 3 Fill” or “Super Loki Launch Countdown” when the CCTK administrator creates it. When a CCTK operator/client starts or attaches to a real-time session, the project file name is selected from a menu of available project configurations. The configuration database identifies and describes the measurements, commands, and interfaces associated with a project.

Because the CCTK administration function is performed off-line from real-time activities, the process of defining a project can be performed independent of the CCTK operational state. This means multiple project configurations can be created or edited during real-time operations.

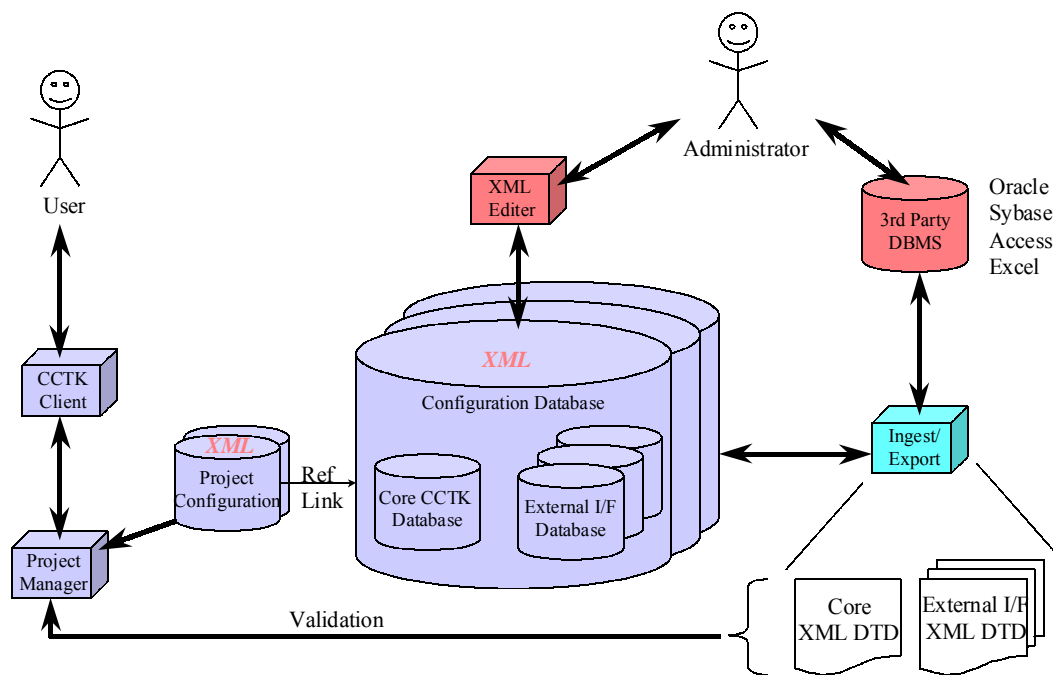


Figure 2-17. Configuration Management Architecture

Figure 2-17 describes the relationship between the project configuration and the configuration database. Because the information elements of CCTK are defined using XML data translation documents (DTD), they can flexibly exchange configuration data with other database management tools.

2.3.4.1.1 Project Configuration

CCTK is comprised of a number of parallel processing real-time tasks that can be configured in various ways. Each of these tasks produce and consume data, commands, messages, and status information using special high speed communications and status resources defined in the project configuration file. When CCTK is activated, the project configuration file is utilized by a special task called “Project Manager” to activate the real-time tasks and allocate all of the resources necessary to operate CCTK in a project configuration.

The Project Configuration is an XML file hierarchically organized as described in Figure 2-18. It contains a project description, startup modes, and CCTK client configuration parameters. The startup mode, or modes, define a reference link to a configuration database, the standard channels, and standard tasks required by CCTK. Modes can be extended or inherited, to create new and/or custom configurations. An example mode extension is the addition of an add-on external interface mode that builds on the “Core CCTK” mode to create the resources and initiate the additional tasks required by the interface. Extended modes are often used to define variations on configuration for activities such as simulation or testing. The modes defined in the project configuration are selectable at run time when CCTK is activated.

The Project Configuration also contains the tree list of applications executable from the CCTK Client. This list is customizable by the system administrator to include references to user-defined displays and applications.

Refer to the *CCTK Administrator's Manual* for more details concerning Project Configuration.

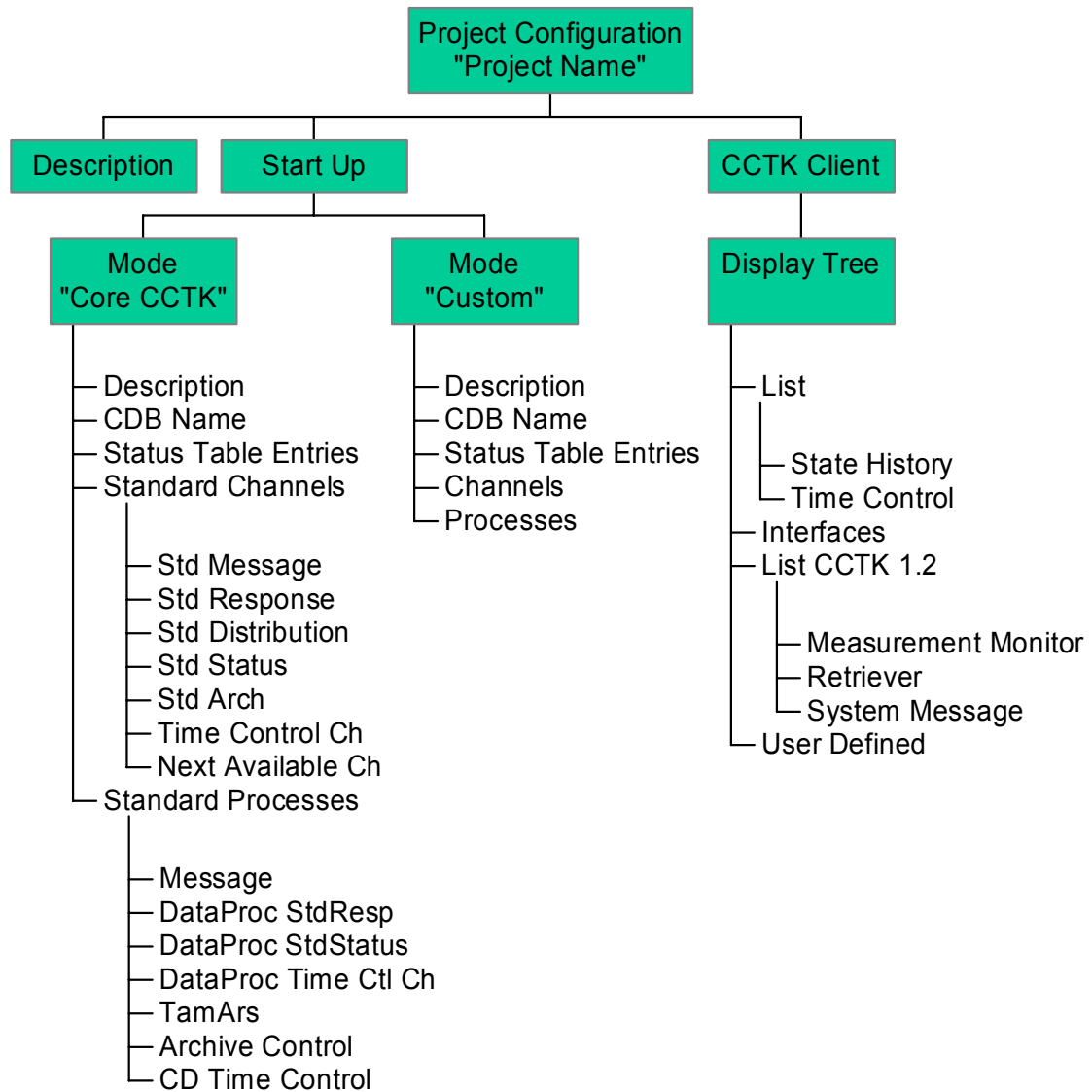


Figure 2-18. Project Configuration Structure

2.3.4.1.2 Configuration Database

The CCTK provides a mechanism for management of project-dependent data and command parameters in a “configuration database” (CDB). The CDB is based on an open XML scheme and provides the ability to create and edit database files off-line from the real-time system software.

New database configurations are easily ingested or exported to/from the common database file formats. Because the CDB is based on XML technology, third party database management systems such as Oracle, Sybase, or Microsoft Access can be integrated with the CCTK configuration management functions.

Just like the Project Configuration discussed in the previous section, the CDB is based on a hierarchically organized XML structure. Contents include definition of interfaces, system notices, and standard CCTK tables. Composition typically includes a “core” CDB that defines the minimum required by a Project, and one or more extended CDB’s that inherit the attributes of the core.

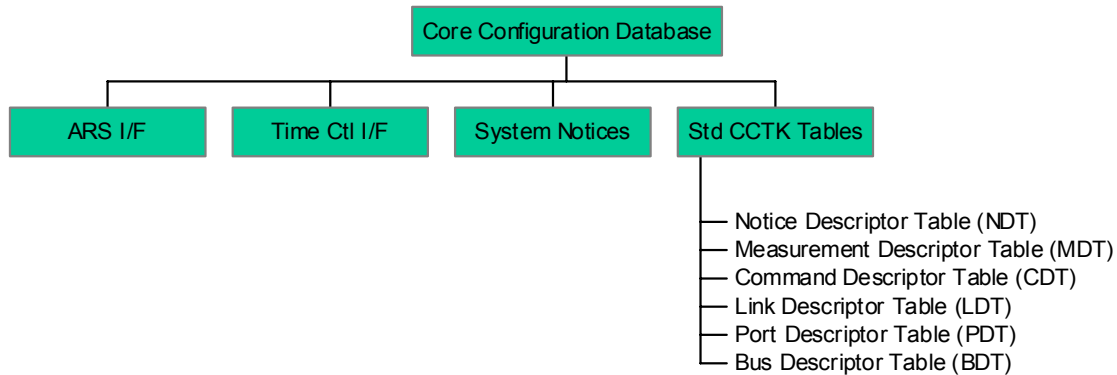


Figure 2-19. Core Configuration Database Structure

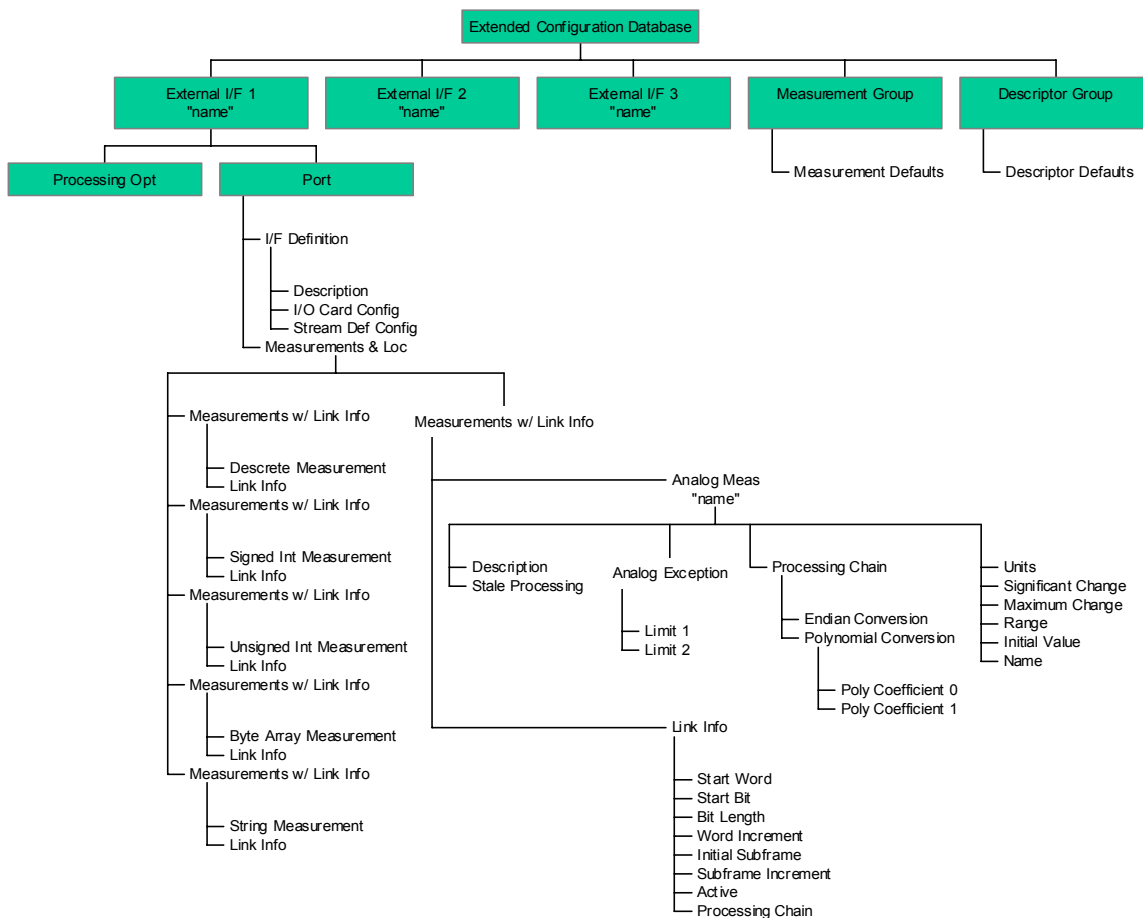


Figure 2-20. Extended Configuration Database Structure

All data and commands in the CDB are assigned a unique symbolic name referred to as “Function Designator” or FD. The FD is the handle by which the CDB and real-time display

and control applications reference specific measurements and commands. The FD symbolic name assignment is instrumental in decoupling application software from the end item hardware and interface protocols. A number of attributes are associated with the FD inside the CDB which tell CCTK where and how to process FD information. For example, an analog measurement FD attributes include:

- Engineering Unit (EU) conversions and linearization
- User Defined Violation/Exception Limits
- Valid ranges, and derivations
- Sample Rate
- Compression Filter Definition
- Stale Processing Characteristics
- Stream/Link/Port Association and Hardware Addressing

System messages, sometimes referred to as “notices,” are also defined in the CDB. The project administrator can customize and extend the message database to use local terminology, or define messages to be posted by user applications.

Like other CCTK real-time tasks, plug-in interfaces are defined in the project configuration file and project configuration database. Interface extensions to the CDB provide the interface protocol specifications needed for the interface processing software. Information such as bus address, port address, bit rate, frame rate, frame structure, sync patterns, and sample rates, are all included as configurable parameters.

Each CCTK interface option comes with a reference manual that provides the administrator with the specific details of how to configure the interface to work with the CCTK core components.

Refer to the *CCTK Administrator's Manual* for more details concerning project configuration.

2.3.4.2 Data Recording and Analysis

CCTK provides a data archiving capability that records all data, commands, events, and system messages in time ordered fashion for later retrieval. CCTK provides client/server access to all archived information via retrieval tools that are accessible to all graphic workstation clients. Separate tasks support the simultaneous continuous real-time data recording, retrieval and viewing of previously archived data as illustrated in Figure 2-21. The user can also display real-time data and view previously archived data simultaneously on one or more workstations.

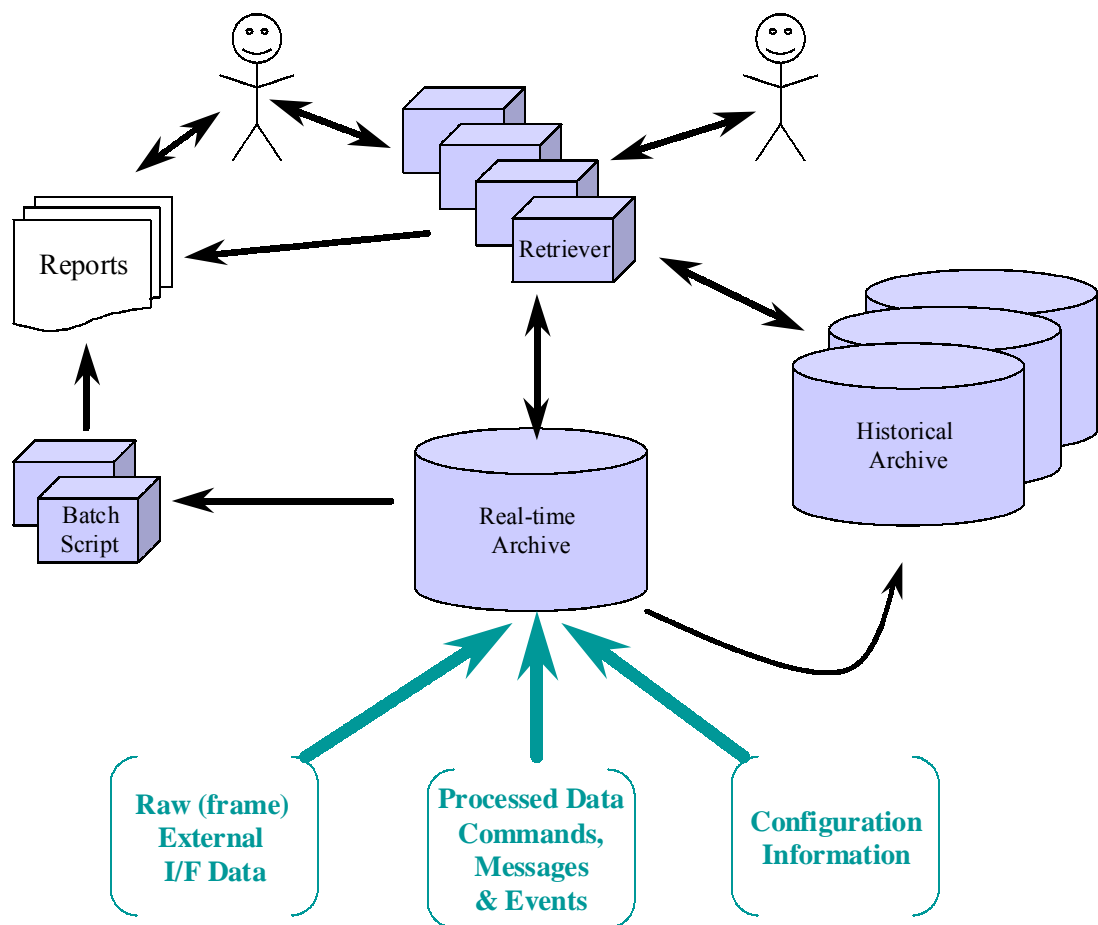


Figure 2-21. Recording and Analysis Architecture

The archive subsystem records digitized information using a special archive file format that is optimized for rapid archive and retrieval access. Even on a modestly configured platform the archive subsystem enables hundreds of thousands of recorded processed data samples to be recorded and retrieved in a few seconds. Archive throughput rates are limited only by the bandwidth of the recording device and media.

The archive subsystem is primarily a listening device; recording all information it receives on its input channel. All data, commands, events, and messages are recorded by default. However, the CCTK administrator can turn off recording for any information source by altering record settings in the CDB. Archive can also be controlled using the CCTK Client.

Retrievals can be initiated from Retriever, a system application, or from a batch processing script. Selection parameters include start time, stop time, list of FD's, FD type (command, measurement, message, etc.) and an array of user-selectable filters that key on specific FD attributes such as exception state, health, and value. Retrieval definitions can be saved to a file for later recall or batch processing. Retrieval results are stored in a file, which can be displayed, printed in tabular form, plotted graphically, or exported to comma delimited (or CSV) file for use in Microsoft Excel and other programs.

2.3.4.3 Peer-to-Peer

CCTK provides peer-to-peer software to allow data to be transmitted between multiple systems over a network interface. The peer-to-peer software allows multiple CCTK systems

to communicate with each other. It also allows CCTK systems to integrate with 3rd party software via the open CCTK peer-to-peer protocol.

Peers exchange information using a standard IP network. The CCTK peer-to-peer software supports both connected and connectionless sockets. When using connectionless sockets, the option is also provided to transmit data using the broadcast or multicast protocols. This lends great flexibility to the CCTK peer processes and allows them to perform many different roles including:

- *Distributing processing load between multiple systems.* For example, data acquisition and decomutation can be performed on one system, the resulting data can be transferred to a second system via peer-to-peer, where data processing and archive is performed.
- *Data distribution to and from multiple custom clients.* For example, if CCTK is being integrated into a legacy system, peer-to-peer software along with a custom client can be used to distribute data between the legacy system.
- *Data request server.* Peer server can be configured to act as a request server. Clients can connect to the server and request measurements that are on the remote system. This works well for clients that need to receive every data sample and have significant processing requirements, such as a graphical strip chart client.
- *Separation of critical tasks.* For example, it may be desired to distribute collected data over the web. However, most web applications are a security concern. Therefore, the peer-to-peer software can be used to move the data to a dedicated web server removing the security concern from the primary data acquisition host.

Figure 2-22 presents the aforementioned examples.

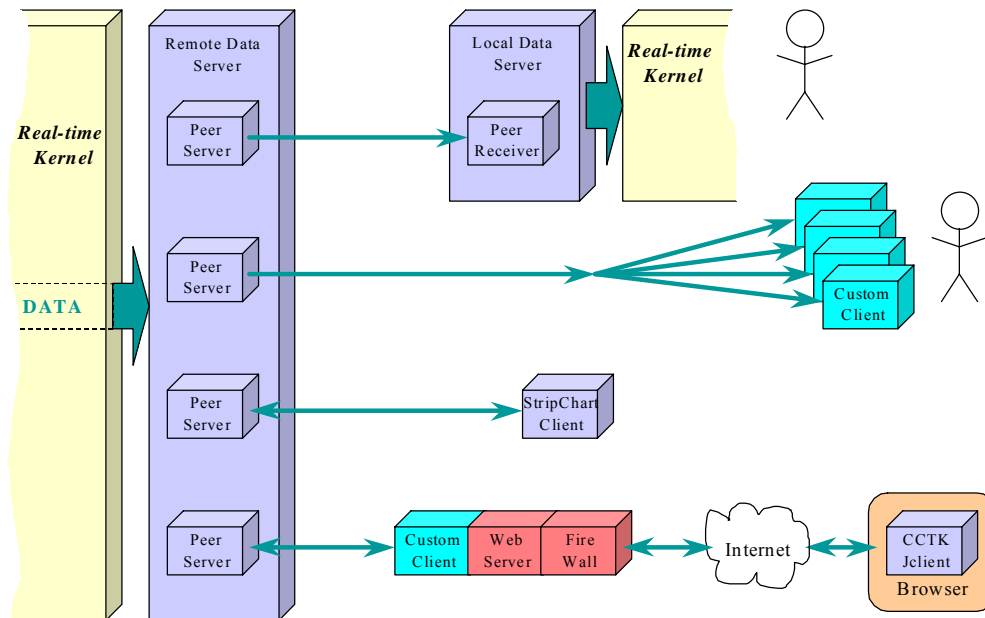


Figure 2-22. Peer Server Architecture

Refer to the *CCTK Administrator's Manual* for details on configuring the peer-to-peer interface.

2.3.4.4 Simulation

The CCTK simulation function provides a flexible means of creating simulations that can scale from simple single sensor manipulation to fully interactive behavioral models. Models created in this environment execute concurrently and work interchangeably with real-time applications and external interfaces as shown in Figure 2-23.

The CCTK simulation capability supports three principal tasks:

- Application development and testing – Provides low level dynamic data simulation for checkout of applications, displays, and configuration databases.
- User training - Provides nominal and non-nominal application-based simulations for operations personnel training. These features also support capabilities for operations research and operations concept exploration.
- System testing and analysis - Provide missing element modeling and configurable project scenarios for system problem resolution, testing, and operational certification.

Simulation models can simultaneously interact with multiple external interfaces, such that an event on one interface can result in a response behavior realized on another interface.

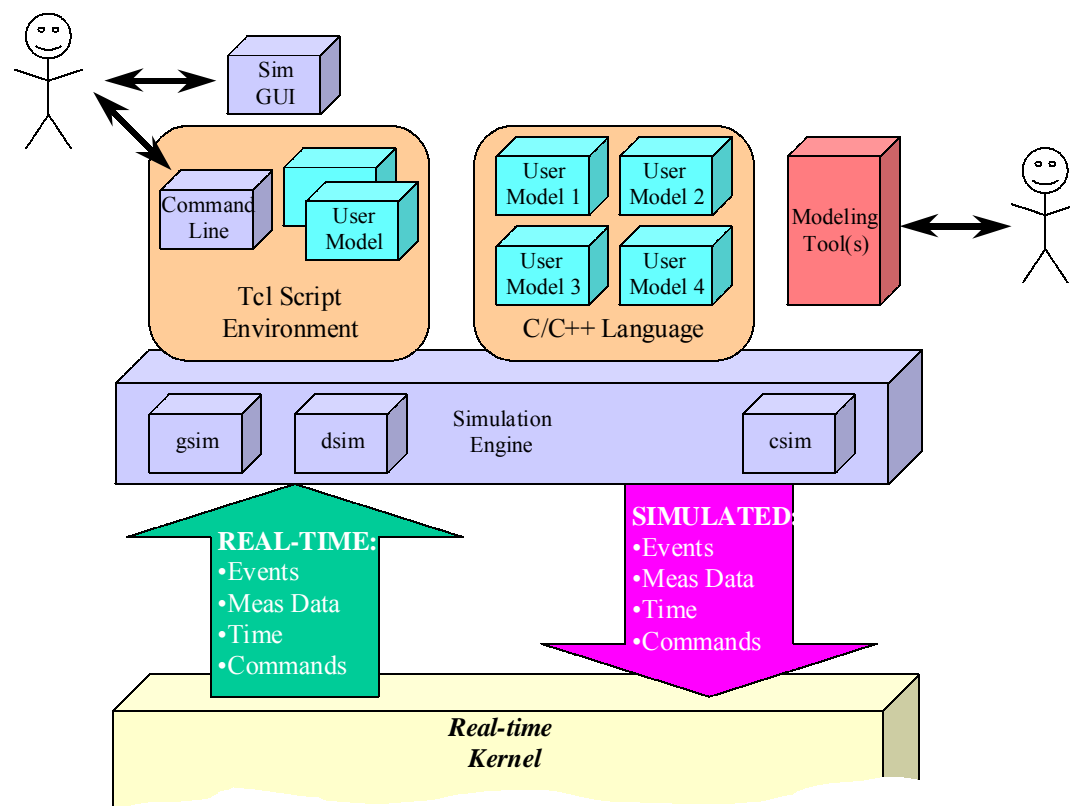


Figure 2-23. Simulation Architecture

Simulations can be created using any of the following:

- **Command Line** - All services provided by the simulation engine are available for use from the command line. This feature is useful for on-the-fly actions such as applying a ramp or sine function to a measurement.
- **Simulator tool** – The simulation tool requires no programming expertise. It allows a user to create basic simulations by defining measurement/command “actors,” actor groups, and discrete event stimuli and responses (asynchronous data events, time events, or user events) using a point and click GUI. The simulation can be stored and reused as a simulation script.
- **Scripts** – All services of the simulation engine are available within the Tcl/Tk scripting package.¹ Tcl provides generic programming facilities such as variables, loops, and procedures that greatly extend the power of the simulation environment.
- **C/C++** – Full native language support provides simulation flexibility only possible with a complete programming language. Also, since C/C++ is the native CCTK system language, performance and response times are optimized for demanding applications not appropriate for the interpretive scripting and command line options.
- **Visual Modeling Tools** – Third party modeling tools such as MatrixXTM and G2TM can be integrated via the CCTK real-time C/C++ API. These tools offer alternative simulation paradigms; however, there are performance limitations associated with these tools that need to be considered when applying them to real-time interactive models.

There are three variations of the simulation engine. They include gsim, which aids development of GLG displays; dsim, which is used for general debugging; and csim, which is integrated with the CCTK real-time kernel.

There is a library of standard simulation modules supported by the simulation engine that includes:

- **Constant** - provides and unchanging value
- **Jitter** - randomly change a constant by a small amount to give the appearance of an actual measurement
- **Sine** - simulate a sine wave
- **Square** - simulate a square wave
- **Saw-tooth** - simulate a saw-tooth wave
- **Expression** - process the expression to determine the value
- **Random** - randomly pick a value from a range
- **Ramp** - change value from a to b over a period then hold value b
- **Combo** - combination of above modules

Detailed capabilities and operation of the CCTK simulation tools are described in the *CCTK Administrator's Manual*.

¹ Tcl/Tk is a freely available, platform independent, language environment developed by John Ousterhout's. Additional information is available at <http://www.neosoft.com/tcl>.

3 CCTK REAL-TIME OPERATION

This chapter describes the user tools and applications available during project execution.

3.1 CCTK Client

CCTK Client is a system application that provides the primary graphical interface for CCTK operation. CCTK Client allows a user to connect to a project, start a project, stop a project, and view graphical displays associated with a project. The individual displays available for a specific project are customizable. CCTK Client uses a multiple document interface (MDI) scheme to present graphical displays to the user. With an MDI interface, multiple windows are controlled within the parent window. The user can open, move, close, and manipulate these windows within the bounds of the parent but cannot move them outside of the parent.

As described in Section 2.1.2, the term “project” describes the thread of activities associated with using CCTK. CCTK uses the term “project” as a generic representation of many different activities that the system can perform. In some cases, a more domain specific term may be appropriate. CCTK allows the project engineer to substitute the term “project” with a more specific term, as appropriate. For example, “mission” could be used for launch operations, or “experiment” could be used for experiments. See the *CCTK Administrator’s Manual* for details on changing the term. Thus it is possible that the word project on all of the screenshots in this section could be replaced with an alternate term.

This section describes CCTK Client when it is attached to the “SIMPLE” project. This project can be found in the CCT home directory under “projects/SIMPLE.” Since the CCTK Client content can differ based upon the current project, it is possible that the screen shots shown in this section will differ from what is seen on-screen if CCTK Client is attached to a project other than SIMPLE.

3.1.1 Overview

Figure 3-1 shows a running CCTK Client attached to a project.

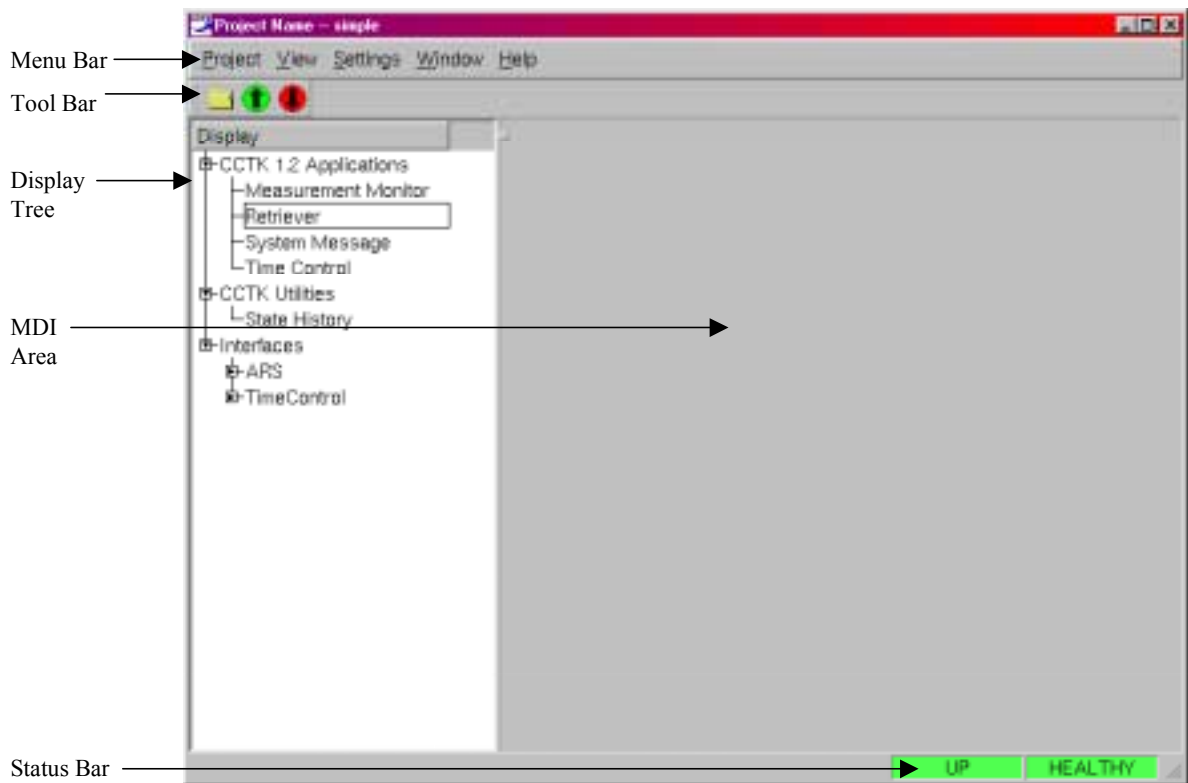


Figure 3-1. CCTK Client

The menu bar at the top of the window allows you to select specific actions for CCTK Client to perform. The toolbar, located below the menu, provides quick access to the more commonly used menu items.

The display tree lists all of the available displays for a user. The display tree is configured on a per project basis and can change from project to project. It is automatically updated when a project is started if the list of available interfaces changes. For more details on the display tree, see Section 3.1.8.

An MDI window contains all of the active displays. You can manipulate windows located within the MDI window. Several menu options are available to allow you to quickly “tile” and “cascade” all of the open windows. More information on using the MDI interface is provided in the Section 3.1.9.

A status bar is located at the bottom of the screen. The status bar shows the project state and health. System states are described in Section 3.1.10.

3.1.2 Obtaining Help

CCTK Client is fully tooltip aware. By resting the mouse on a screen area for several seconds, a simple tooltip will be displayed giving a brief description of the area where the mouse is located. It is also possible to obtain more detailed help information by selecting the “What’s This” option from the help menu. Upon selecting “Help→What’s This” the cursor changes to a pointer and question mark. By clicking on an area within CCTK Client, more detailed help

will be displayed. Figure 3-2 shows one of the CCTK Client dialogs with What's This help displayed.

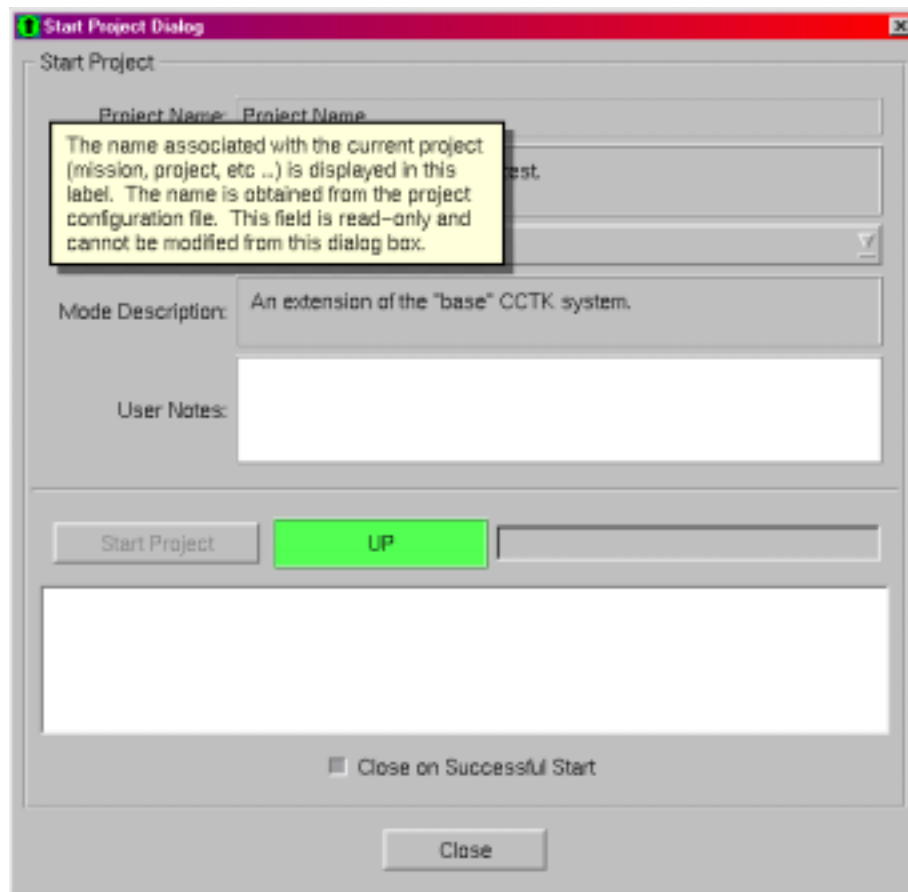


Figure 3-2. CCTK Client, What's This Example

3.1.3 Opening a Project

The CCTK architecture allows multiple projects to exist on a single system. Therefore, CCTK Client allows you to select which project you would like to interact with. CCTK Client may have only one project open at a time.

When a project is selected, the project name appears in the title bar of the CCTK Client window. The state and health of the project appears in the status bar. If CCTK Client does not have a project opened, the state and health fields in the status bar will contain the strings “(not attached)” and “(n/a)” respectively. Figure 3-3 shows the CCTK Client status bar when it is not attached to a project.

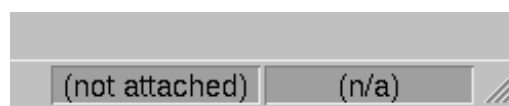


Figure 3-3. CCTK Client Status Bar, not attached to a project

Figure 3-4 shows the CCTK Client status bar when it is attached to a project, the project is UP, and the health is HEALTHY.



Figure 3-4. CCTK Client Status Bar, attached to a project

If no project is open, or you would like to open a different project, select the open project action. This can be done in one of two ways. You can either select Project→Open... from the menu or click on the Open Project icon in the toolbar. Figure 3-5 highlights the CCTK Client Open Project button on the toolbar. You can also use the open project key accelerator sequence Ctrl+O.

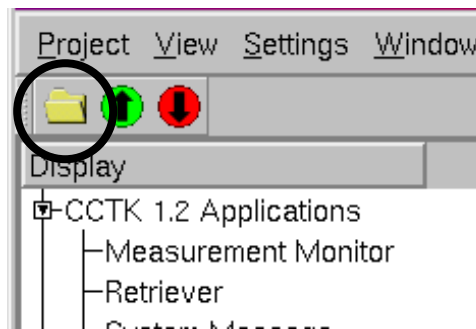


Figure 3-5. CCTK Client Toolbar, Open Project Button

Upon selection of the open project action, the open project dialog box shown in Figure 3-6 is displayed. This modal dialog box allows you to preview key project parameters.

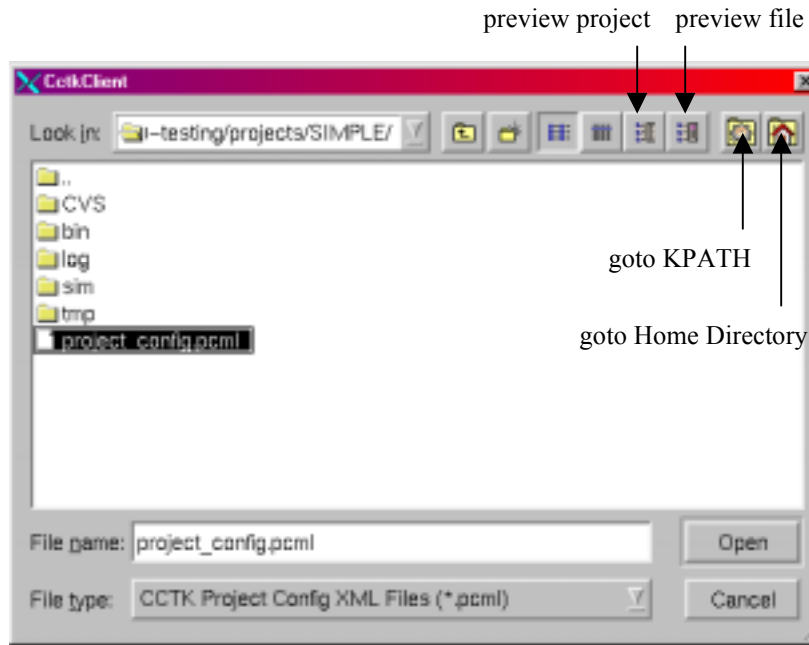


Figure 3-6. CCTK Client, Open Project Dialog

Select a project by clicking on a CCTK project configuration file in the file selection tree. Typically, CCTK project configuration files end in a “.pcm1” suffix. The standard name for a CCTK project configuration file is “project_config.pcm1.” The project configuration file is an XML file that contains information on the configuration of a CCTK project. The *CCTK Administrator’s Manual* provides additional information on managing this file.

By clicking on the preview project button, the dialog box opens a preview project window. By clicking the preview file button, the dialog box opens a text viewer and displays the selected project configuration file.

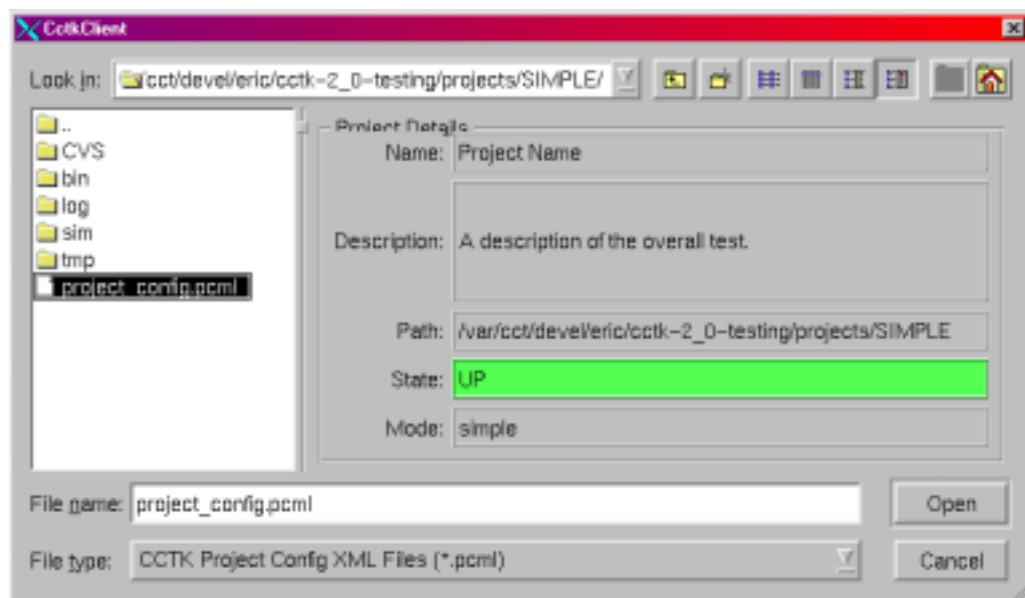


Figure 3-7. CCTK Client, Open Project Dialog with Preview Project Details Shown

Figure 3-7 above shows the Open Project dialog with the preview window open. This window shows information such as project name, project description, current state, and current mode of the project.

Once a project is selected, click the Open button on the lower right side of the dialog box to open the project. The open action can be canceled by clicking the Cancel button. If a project is already open within CCTK Client, it will be closed prior to opening the new project.

3.1.4 Closing a Project

Although this feature is rarely used, it is possible to close the current project by clicking on the “Project→Close” menu item in CCTK Client.

3.1.5 Starting a Project

If the opened CCTK project is in the “DOWN” state, it can be started from the CCTK Client. Only privileged users may start a CCTK project. A project can be started by selecting “Project→Start” from the menu or clicking on the Start Project icon in the toolbar. Figure 3-8 highlights the open project button on the CCTK Client toolbar. The start project key accelerator sequence is Ctrl+S.

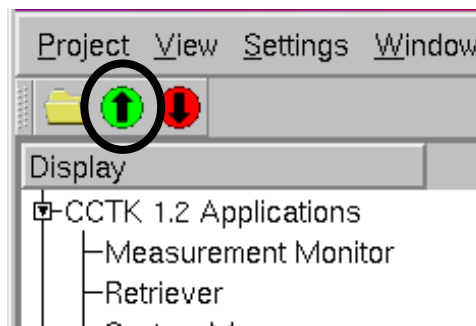


Figure 3-8. CCTK Client Toolbar, Start Project Button

The start project dialog box provides detailed information on starting a project. Figure 3-9 shows the CCTK Client Start Project Dialog.

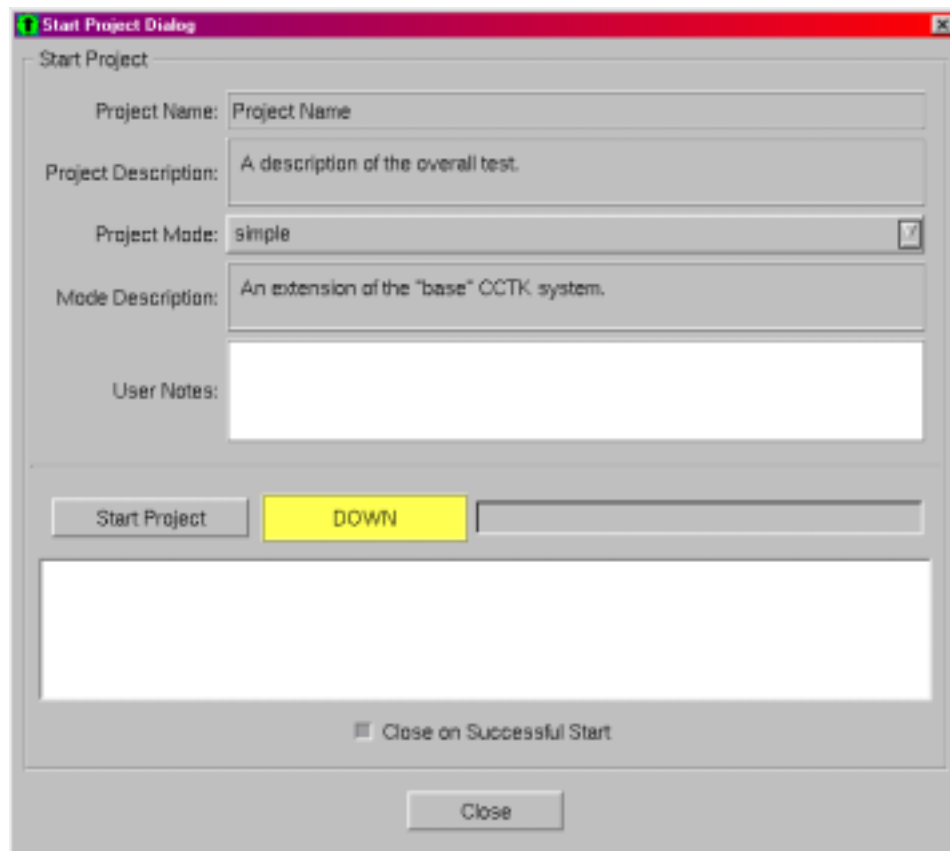


Figure 3-9. CCTK Client, Start Project dialog box

The start project dialog displays the following information about the project:

- The project name
- The project description
- All of the modes available for this project via a drop down list
- The description of the mode selected in the drop down list

The above information cannot be changed from this dialog box. To change the project configuration file must be modified. See the *CCTK Administrator's Manual* for information on modifying the project configuration file.

Before starting the project, an initial mode can be specified. If an initial mode is not specified, the default mode is automatically selected. This default mode may be appropriate a majority of the time.

Part of the task of starting a project includes specification of user notes. User notes are a free form description of why the user is starting the project. The user notes are not required. The user notes become part of the historical record of project execution and can be viewed in the state history display.

Click on the start project button to start the project as shown in Figure 3-8. The current state of the project is displayed for reference. A progress bar displays the percentage of startup completed. A scroll window displays textual information as the project starts. As the project starts up, the state label will change to "STARTING_UP." If the project successfully starts,

the state label will turn green and “UP” will be displayed. If the startup fails, the state label will turn red and “STARTUP_FAILED” will be displayed. See the *CCTK Administrator’s Manual* for troubleshooting startup problems.

The check button at the bottom controls whether or not the start project dialog box should automatically close after a successful start. The dialog box can be manually closed at any time using the Close button. Closing the dialog box does not cancel any requested start up of the system.

3.1.6 Stopping a Project

If the project is in the “UP” state, it can be stopped from the CCTK Client. Only privileged users may stop a CCTK project. A project can be stopped by selecting “Project→Stop” from the menu or clicking on the Stop Project icon in the toolbar. Figure 3-10 highlights the stop project button on the CCTK Client toolbar. The stop project key accelerator sequence is Ctrl+T.

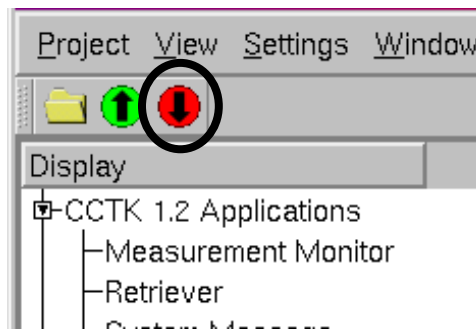


Figure 3-10. CCTK Client Toolbar, Stop Project

The stop project dialog box provides detailed information on stopping a project. Figure 3-11 shows the stop project dialog box.

User notes may be entered before stopping the project. The user notes are a free form description of why the user is stopping the project and are not required. As with starting a project, the user notes become part of the historical project record and can be viewed in the state history display.

Normally, a CCTK project can only be stopped when it is in the “UP” state. However, under rare circumstances it is possible that problems occurred while starting/stopping a project. In this case it is necessary to stop a project even though it is not in the “UP” state. The “Shutdown CCTK System Regardless of Current State” checkbox is used to force a shutdown. This checkbox should typically be left unchecked. It is only needed when problems arise during start up of the system.

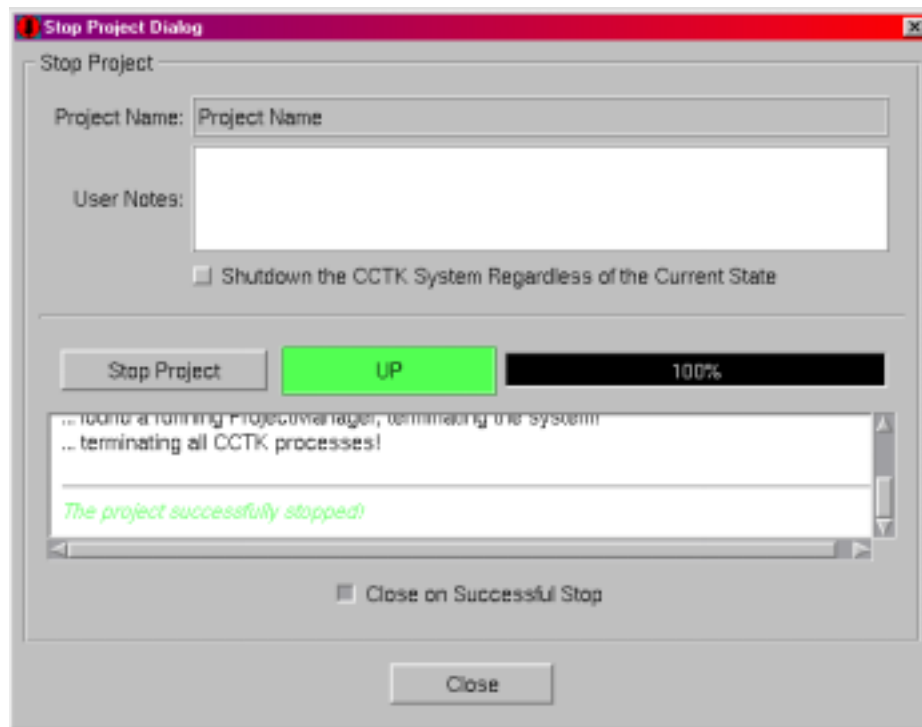


Figure 3-11. CCTK Client, Stop Project dialog box

To stop the project, click on the stop project button. The current state of the project is displayed for reference. A progress bar displays the percentage of shutdown completed. A scroll window displays textual information as the project stops. As the project shuts down, the state label will change to “SHUTTING_DOWN.” If the project successfully stops, the state label will typically display “DOWN.” If a shutdown was forced, then the state label will display “FORCED_DOWN” (see Table 2-6 on page 23 for description of CCTK states).

The check button at the bottom controls whether or not the stop project dialog box should automatically close after a successful stop. The dialog box can be manually closed at any time using the “Close” button.

3.1.7 Exiting CCTK Client

Selecting the exit action will terminate CCTK Client. The exit action can be selected from the “Project→Exit” menu item. The exit action may also be selected using the exit project key accelerator sequence Ctrl+X. It is important to note that exiting CCTK Client does not affect the state of a project. Even if the user that started the project exits, or all users associated with a project exit CCTK Client, the project will continue to run.

3.1.8 Display Tree

The display tree provides the primary navigation control for the CCTK Client window. From the display tree, you can select the displays you would like to view. The display tree is always displayed on the left side of the CCTK Client window. It resides between the toolbar and the

status bar. The display tree can be hidden or shown by selecting “View→Display Tree” from the menu.

The content of the display tree is controlled by the project configuration file. Please see the *CCTK Administrator's Manual* for information on modifying the project configuration file.

To select a display, single click on the display in the display tree. Once selected, the display will be shown in the MDI window to the right of the display tree.

3.1.9 Multiple Document Interface Window

The multiple document interface window shows all of the displays selected for viewing. Each display is managed as an independent window. Each display can be manipulated within the bounds of the CCTK Client window using basic window navigation controls.

Two menu options operate on the MDI window. By selecting “Window→Tile” all selected displays will be resized and moved to fit within the available space with no window overlapping. By selecting “Window→Cascade” all selected displays will be resized and moved in a cascade style, title bars lined up, windows overlapping. Figure 3-12 shows an example of tiled windows in CCTK Client. Figure 3-13 shows an example of cascaded windows in CCTK Client.

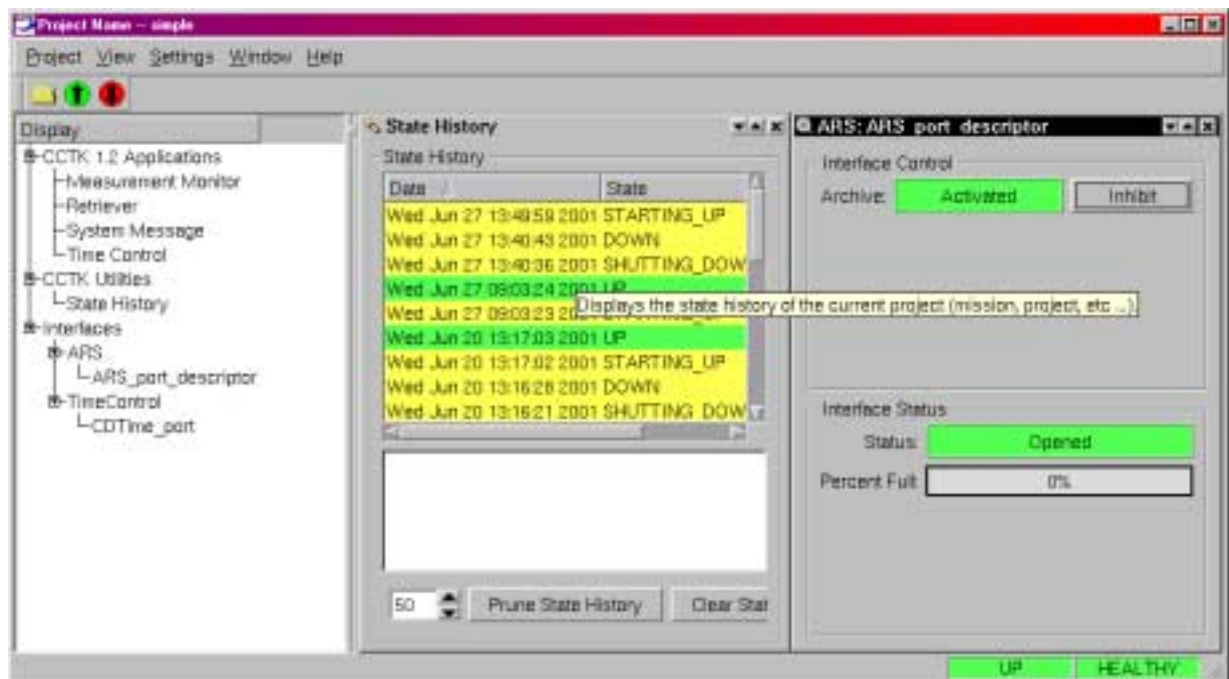


Figure 3-12. CCTK Client MDI Interface, Tiled Windows

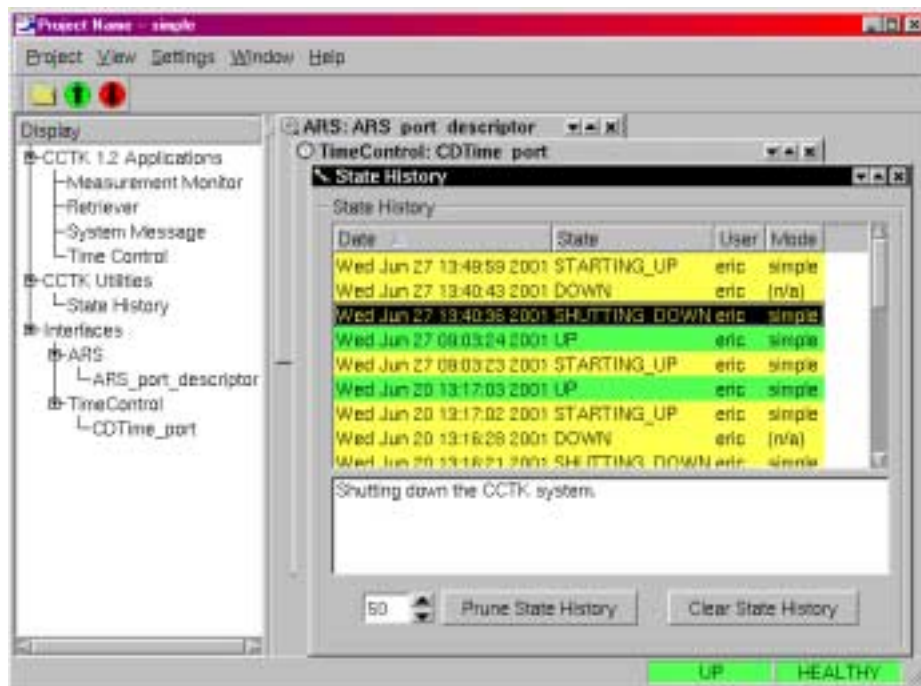


Figure 3-13. CCTK Client MDI Interface, Cascaded Windows

3.1.10 Status Bar

The status bar located at the bottom of the screen provides general status feedback. The project state and project health are displayed on the very right side of the status bar.

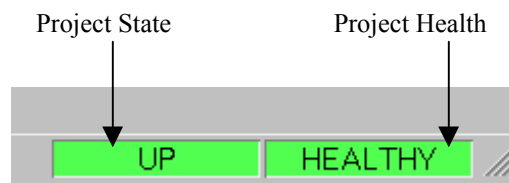


Figure 3-14. CCTK Client, State and Health Indicators

Project state indicates the current state of the project that is open within the CCTK system. Project health indicates the current health of a running CCTK system. Project health is only valid when the system is in the “UP” state. Please see Section 2.3.1.1.4 in the Theory of Operation chapter for more information on project health and status.

3.1.11 Individual Displays

CCTK Client displays individual CCTK displays in the MDI window. CCTK has a set of prebuilt displays that are distributed with the product. You can also create displays that can be integrated into CCTK Client. See the *CCTK Developer's Manual* for more information on creating custom CCTK displays. The list of displays available to an individual user is controlled by the project configuration file and can vary between projects. Because of the

level of customization available, the section will not be accurate for all installations of CCTK. If you find discrepancies between this section and what is available, please consult your system administrator.

Displays are typically tied to the state of the CCTK system. Some displays, such as Archive Control, are only valid if the system is in the “UP” state. Therefore, Archive Control is desensitized if the project is not “UP.” On the other hand, some displays are valid regardless of the state of the system. State History is an example of a display that operates whether the project is “UP,” “DOWN” or something else.

The following list shows the standard CCTK displays and provides a short description of each. Additional details can be found in the sections describing each of these displays.

- **State History:** State history presents a history of project executions. State history shows when a state transition occurred, which user initiated the transition, and any user notes applicable to the transition.
- **Archive Control:** Archive Control provides detailed control over the CCTK archive subsystem. It will only be available if the selected project is configured with an archive subsystem.
- **Time Control:** Time Control provides detailed control over the CCTK time control subsystem. It will only be available if the selected project is configured with a time control subsystem.

3.1.11.1 State History

The State History display provides a detailed list of the state transitions of a CCTK project. Each time a CCTK project transitions from one state to another (i.e. from “DOWN” to “STARTING_UP”), the transition is recorded. The State History display allows the user to view and manage the transition history. The following information is recorded about each state transition:

- **Date** – indicates the date when the state transition occurred.
- **User** – indicates the user that initiated the state transition.
- **State** – indicates the state that was transitioned to.
- **Mode** – indicates the mode (only valid when associated with a state that involves a running project, i.e., STARTING_UP, UP, or SHUTTING_DOWN).
- **User Notes** – indicates the user notes entered by the user for “STARTING_UP” and “SHUTTING_DOWN” or the notes applied by the system for automated state transitions.

Figure 3-15 shows the state history display. This display is divided into three main sections. The “state history” list box lists all of the state transitions in a tabular format. It can be sorted by clicking on the column headers. The user notes are shown for the selected entry in the state history list in the area below the State History list. Only one entry may be selected at one time. The “STARTING_UP” and “SHUTTING_DOWN” states list the notes entered by the user in the start/stop dialogs. The other states list user notes generated by the system. The control section allows the user to clear old state history entries.

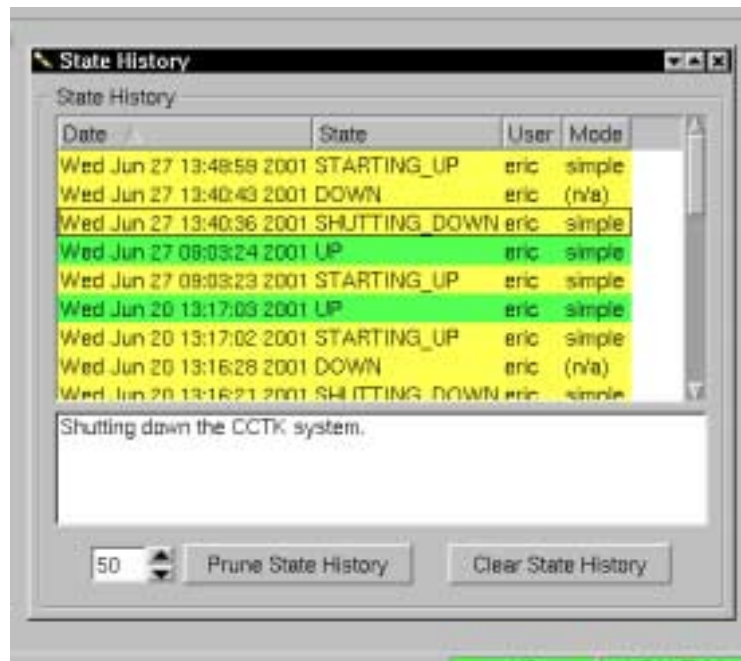


Figure 3-15. CCTK Client, State History Display

You have two options for clearing old state history entries. You may clear all state history entries by clicking the “Clear State History” button or prune the state history to a specific number of entries. Enter the number of entries to leave in the spin box and click on the “Prune State History” button to clear a certain number of entries. You must have the appropriate permissions to modify the state history. It is possible to alter the state history by editing the state history XML file. Please see the *CCTK Administrator’s Manual* for more information on the system state file.

3.1.11.2 Archive Control

Archive Control displays the status of the archive subsystem and providing some basic controls. Archive Control will only be present in the display tree if an archive subsystem is configured for the selected project. Consult the project administrator to see if the current project is configured with an archive subsystem. Figure 3-16 shows the archive control interface.

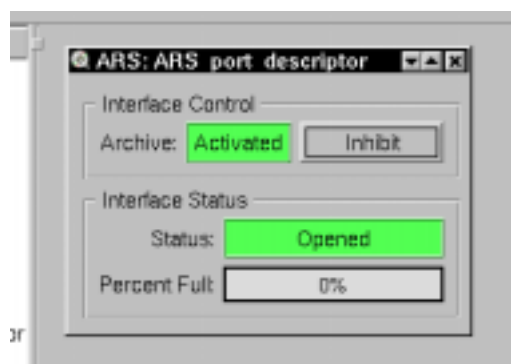


Figure 3-16. CCTK Client, Archive Control Display

The archive subsystem in CCTK can be in one of two states, inhibited or activated. When inhibited, the archive subsystem is receiving data, but is not recording it. When activated, the archive subsystem is receiving data and recording it to disk.

The archive status can be one of three values: open, closed, or disk error. “Open” indicates that the archive is opened and that data is being written.” Closed” indicates that the archive is closed and that no data is being written. “Disk error” indicates that an error has occurred while attempting to write data to the archive.

The archive may be activated/inhibited at any time while the project is operating. The “percent full” field shows the amount of archive space used. Once the archive is full, the archive subsystem will automatically close the archive and stop archiving data.

3.1.11.3 Time Control

Time Control provides detailed control over the countdown time in CCTK. Time Control allows you to set the countdown time as well as schedule start and stop events. Time Control will only be present in the display tree if a time control subsystem is configured for the selected project. Consult the project administrator to see if the current project is configured with Time Control. Figure 3-17 shows the Time Control window.

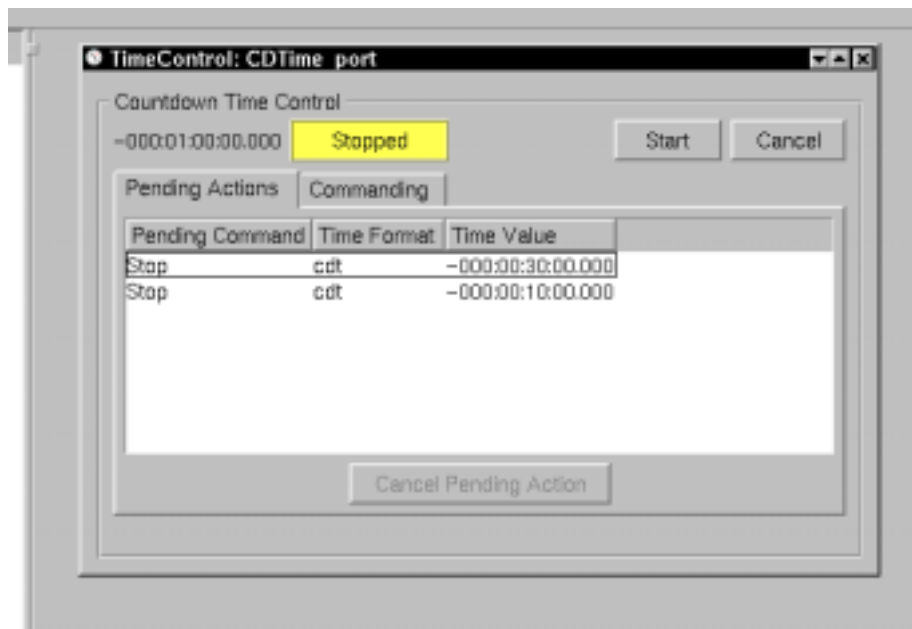


Figure 3-17. CCTK Client, Time Control Display

The current countdown time and countdown status are displayed near the top left of the window. Buttons to Start/Stop and Set/Cancel are present at the top right of the window. These buttons are sensitive to state and will alter their operation based upon the current state of the countdown clock. Table 3-1 summarizes the different states of these buttons.

Table 3-1. CCTK Client, Time Control Button States

Countdown State	Left Button Operation	Right Button Operation
Cancelled	(desensitized)	Set
Stopped	Start	Cancel
Running	Stop	(desensitized)

When the countdown clock is in the cancelled state, it can be set by clicking on the “Set” button in the upper right corner of the window. Figure 3-18 shows the dialog box used to set the countdown clock.

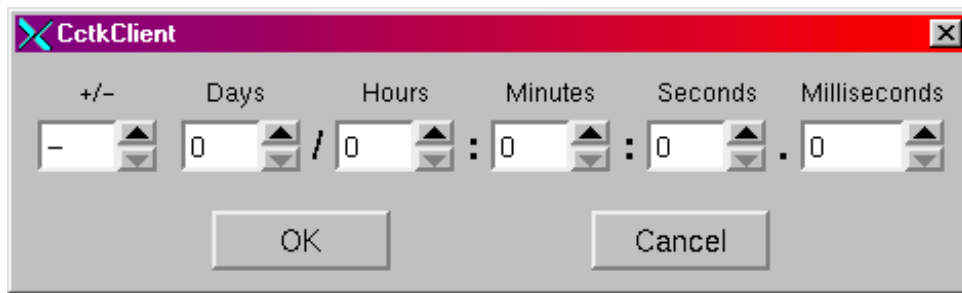


Figure 3-18. CCTK Client, Time Control Set CDT Dialog

To set the countdown clock, use the spin box controls to change the countdown time to the desired values or enter the desired values directly into the text entry boxes. To accept the entered values, click OK. The status of the countdown clock should update to “Stopped” and the entered countdown time should be displayed on the main Time Control window.

It is possible to schedule actions that will occur at some point in the future. For example, it is possible to schedule an action that will “stop” the clock at countdown time T-minus one minute and thirty seconds. The scrolled list in the lower section of the Time Control window lists these pending actions. It is possible to cancel a pending action by selecting it in the list and clicking the “Cancel Pending Action” button.

Scheduling countdown time actions is accomplished through the commanding pane. The commanding pane is a generic interface to the different commands that can be sent to the countdown time control subsystem. Figure 3-19 shows the commanding pane.

When using the commanding pane, first select the type of command you wish to send, where type is one of Set, Start, Stop, or Cancel. Next, select the option associated with this command. The option argument is only selectable when the Cancel type is selected. When the Cancel type is selected, the option argument indicates the type of command you want to Cancel (Set, Start or Stop). Next, the time format is selected. The time format can be one of Now, CDT, UTC, or Local. The time format indicates when the command should take place. Finally, if CDT, UTC, or Local time format is selected, an appropriate time must be entered into the time entry box. Figure 3-19 shows an example of the UTC/Local time entry box. The CDT time entry box in the commanding pane is identical to the CDT entry box in the set countdown time dialog box. Table 3-2 provides a summary of the different commanding options.

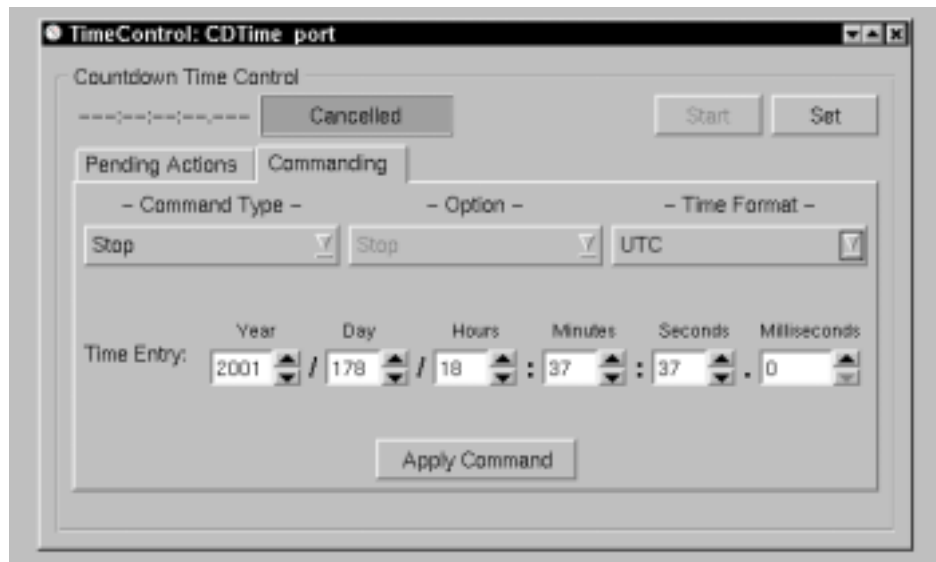


Figure 3-19. CCTK Client, Time Control Commanding Details Pane

Table 3-2. CCTK Client, Time Control CDT Command States

Command	Option	Time Format	Description
Set	Set	Now	Sets the countdown time to 0.
Set	Set	CDT	Sets the countdown time to the user specified CDT time based upon the CDT entry box.
Start	Start	Now	Starts the countdown clock immediately.
Start	Start	Local	Creates a pending action that starts the countdown clock at the specified local time. Only one pending start command may be active at any given time.
Start	Start	UTC	Creates a pending action that starts the countdown clock at the specified UTC. Only one pending start command may be active at any given time.
Stop	Stop	Now	Stops the countdown clock immediately.
Stop	Stop	CDT	Creates a pending action that stops the countdown clock at the specified CDT. Multiple pending stops may be active at the same time.
Stop	Stop	Local	Creates a pending action that stops the countdown clock at the specified local time. Multiple pending stops may be active at the same time.
Stop	Stop	UTC	Creates a pending action that stops the countdown clock at the specified UTC time. Multiple pending stops may be active at the same time.
Cancel	Set	Now	Cancels the current countdown clock.
Cancel	Start	(n/a)	Cancels the pending start command if one exists.
Cancel	Stop	CDT	Cancels the pending stop command that matches the given CDT.
Cancel	Stop	Local	Cancels the pending stop command that matches the given local time.
Cancel	Stop	UTC	Cancels the pending stop command that matches the given UTC time.

3.2 System Message Display

System Message GUI displays the system messages generated by the current CCTK project. System Message GUI can be customized using configurable panes (a scrolled list of messages) and filters (the ability to limit the system messages received) to meet the needs of many different users. System Message GUI is a stand-alone application that is not contained within CCTK Client.

3.2.1 Overview

By default, System Message GUI is configured with a single pane that displays all system messages generated by the current CCTK project. Figure 3-20 shows the basic System Message GUI.

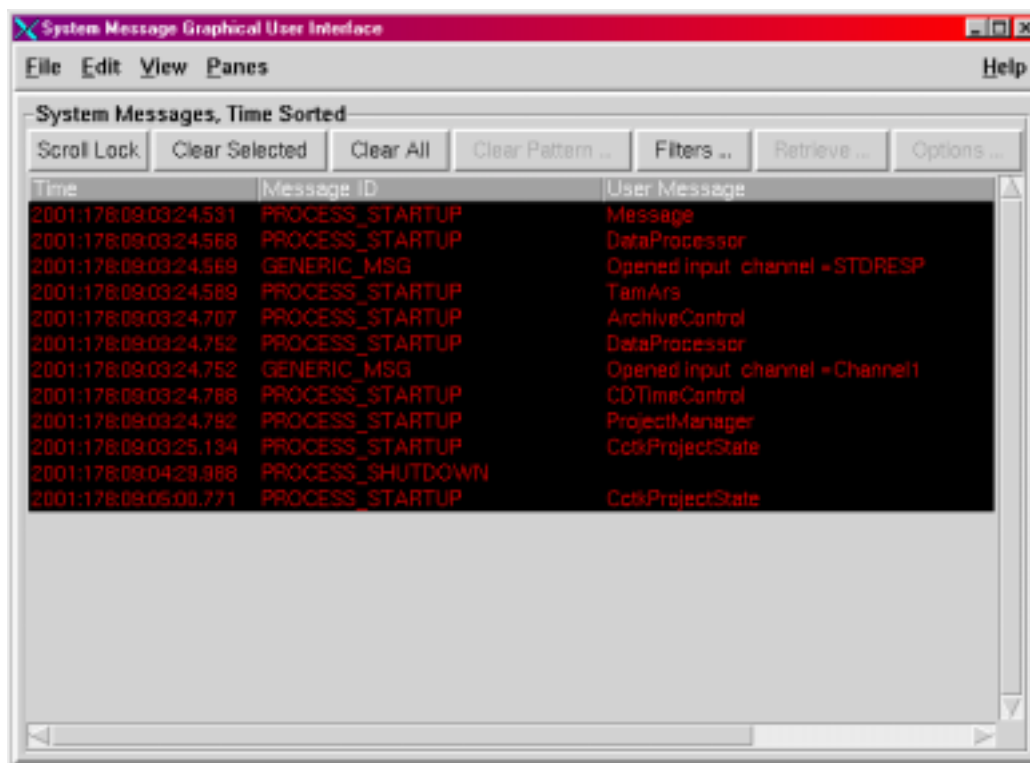


Figure 3-20. System Message GUI, Time Sorted View

Each time a new system message is generated within CCTK, it is appended to the system message list. This view will probably be adequate for most users needs.

It is also possible to operate System Message GUI in a mode with three panes, one for each level of criticality: Critical, Cautionary, and Informational. Figure 3-21 shows this version of System Message GUI.

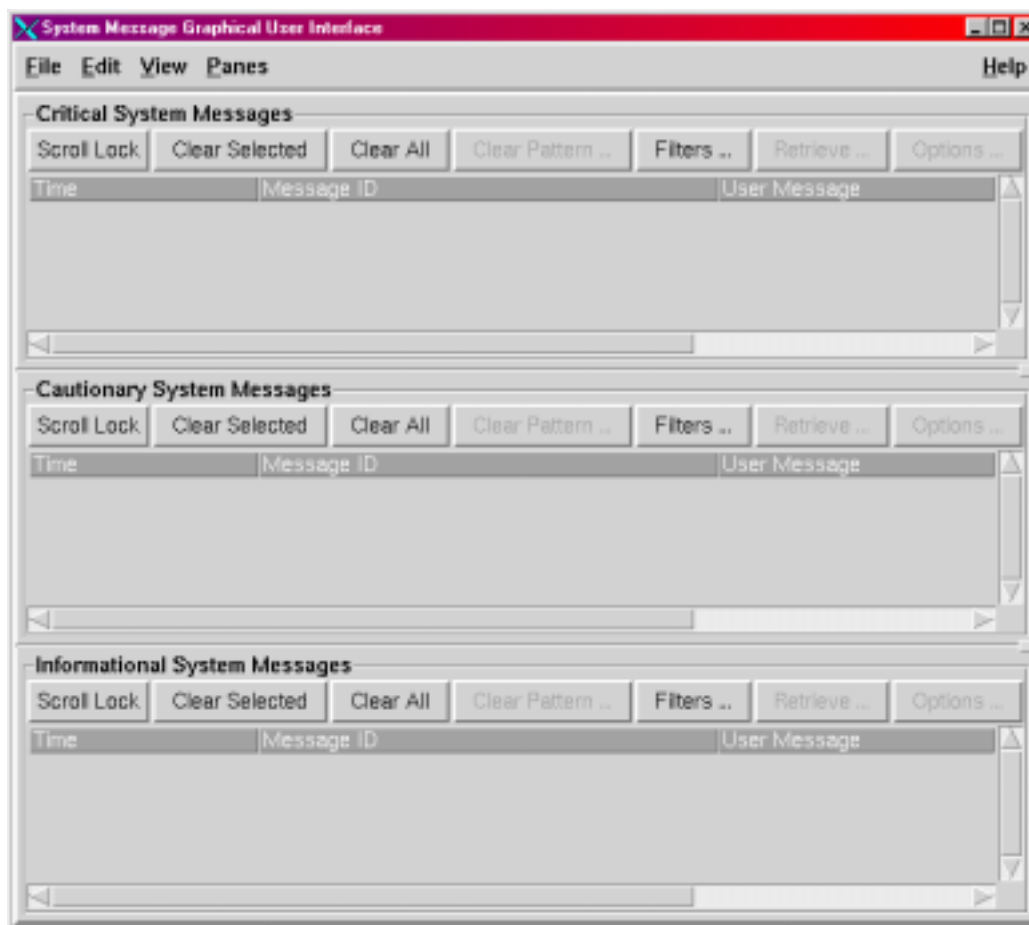


Figure 3-21. System Message GUI, Criticality Sorted View

You can alternate between the single pane, time sorted view, and the three pane criticality sorted view by selecting the “View→Set Predefined→Single Pane, Time Sorted” menu item and the “View→Set Predefined→Three Pane, Criticality Sorted” menu item respectively. Whenever the System Message GUI view is switched from one to the other, all system messages displayed are lost. Past system messages can be retrieved using the archive subsystem.

3.2.2 Clearing Messages from a Pane

It is possible to remove old messages from a pane. All messages may be removed from a pane by clicking on the “Clear All” button on the associated pane. Selected messages may be removed from a pane by clicking on the “Clear Selected” button on the associated pane. At least one message must be selected before this button is sensitized.

3.2.3 Scroll Lock

Each pane will automatically scroll down as new messages are added to the pane. Automatic scrolling can be disabled by clicking the “Scroll Lock” toggle button in each pane. When

“Scroll Lock” is released, the pane will scroll normally. When “Scroll Lock” is pressed and held, the pane will not scroll.

3.2.4 Filters

Each pane has an associated set of filters. Before a system message is displayed in a pane, it is checked against the pane’s filters to see if it should be ignored. You can modify the filters associated with a pane by clicking on the “Filters ...” button. The “Edit Filter” dialog box shown in Figure 3-22 will be displayed.

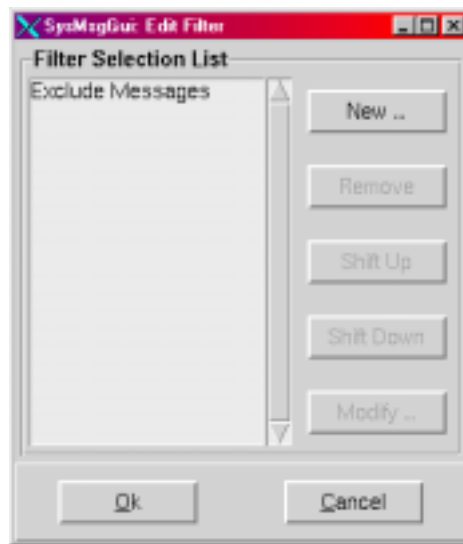


Figure 3-22. System Message GUI, Edit Filters Dialog

On the left side of the dialog box is the list of filters associated with this pane. On the right side of the dialog box are the controls used to manage the filters. To add a new filter, click the “New...” button. A dialog box will pop up prompting for the name of the new filter. Once the filter is named, it will be inserted into the “Filter Selection List.” An existing filter can be removed by selecting it in the list and clicking the “Remove” button.

Because of the way the filter algorithms work, order of the filters is important. Therefore, it may be necessary to change the order of the filters. The “Shift Up” and “Shift Down” buttons allow the ordering of the filters to be changed. To change the order of a filter, select one of the filters in the “Filter Selection List” and click the “Shift Up” or “Shift Down” buttons.

In addition, the filter details can be modified by selecting a filter in the “Filter Selection List” and clicking the “Modify...” button. Another dialog box will be displayed, this one showing the details associated with the filter. Figure 3-23 shows the “Edit Filter Details” dialog.

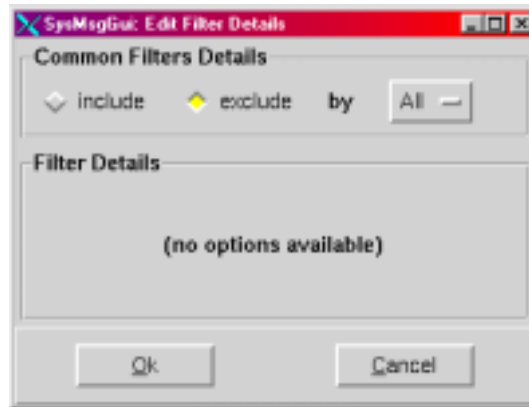


Figure 3-23. System Message GUI, Edit Filter Details Dialog

There are two primary kinds of filters:

- Include filters: If the system message matches an include filter, it will be inserted into the pane and no further filters will be checked.
- Exclude filters: if the system message matches an exclude filter, it will be not be inserted into the pane and no further filters will be checked.

Once all filters have been checked and no matches have been found, the system message will be inserted into the pane by default.

Once a filter is marked as either include or exclude, the filter parameter needs to be selected. The possible filter parameters are:

- Message ID
- Criticality
- All

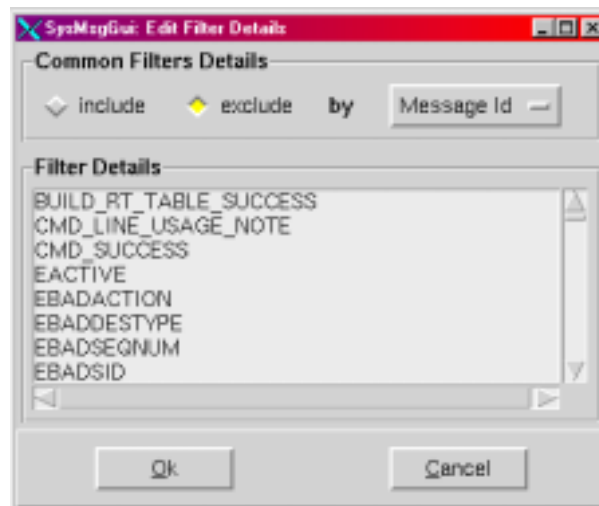


Figure 3-24. System Message GUI, Edit Filter Details by Message ID Dialog

When filtering by “Message Id” a list of all system messages is displayed. When filtering by “Criticality” a list of valid criticality values is displayed. Individual items can be selected from these lists for inclusion or exclusion based upon the type of filter. Figure 3-24 and

Figure 3-25 show the “Filter Details” dialog box with “Message Id” and “Criticality” selected respectively.

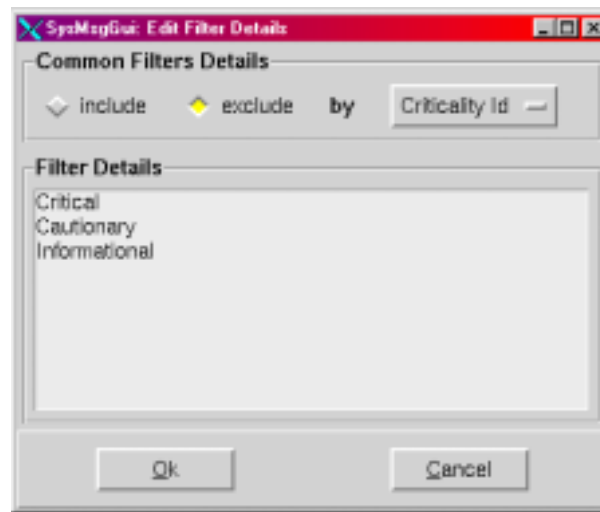


Figure 3-25. System Message GUI, Edit Filter Details by Criticality Dialog

Several examples of defining filters are presented below.

3.2.5 Working with Panes

It is possible to add additional panes to the system message GUI. To create a new pane, select the “Panes→New...” menu item. A dialog box will appear asking for the new pane's name. Enter the name in this dialog box, click “Ok” and the pane will appear. When a new pane is created, no filters are defined. Since no filters are defined, all system messages are included by default so all system messages will be displayed in the pane. Figure 3-26 shows the new pane dialog box.

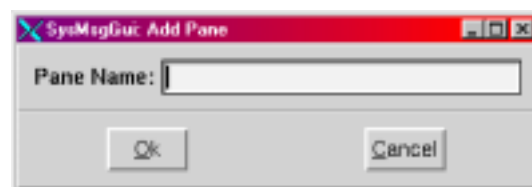


Figure 3-26. System Message GUI, Add Pane Dialog

To remove an existing pane, select “Panes→Remove” from the menu. Another submenu will appear listing all available panes. Simply select the pane that you wish to remove.

It is also possible to show/hide panes. To hide a visible pane, select “Panes→Hide” from the menu. Another submenu will appear listing all visible panes. Simply select the pane that you wish to hide. To show a hidden pane, select “Panes→Show” from the menu. Another submenu will appear listing all of the hidden panes. Select the pane that you wish to show.

3.2.6 A Practical Example

Suppose you are a flight engineer responsible for monitoring the electrical subsystem on a space launch vehicle. In this position, you wish to be notified of the following items:

- All critical system messages
- All messages associated with the electrical subsystem.

In addition, you wish to view these two sets of messages in different lists. The following steps will setup System Message GUI to display only the messages of interest after System Message GUI has been started:

1. Remove all existing panes
2. Create a new pane called “Critical Messages”
3. Select “Filters...” associated with this new pane
4. Create a new filter called “Exclude Criticality”
5. Modify this filter
6. Select “exclude” by “Criticality”
7. Select “Cautionary” and “Informational” from the list.

This completes setup of the first pane. All messages of criticality “Cautionary” and “Informational” will be excluded while “Critical” messages (all that is left) will be included by default. Continuing with the setup of the next pane:

- Create a new pane called “Electrical Messages”
- Select “Filters...” associated with this pane
- Create a new filter called “Include Messages”
- Modify this filter
- Select “include” by “Message”
- Select ALL of the messages associated with the electrical subsystem
- Create a new filter called “Exclude All”
- Modify this filter
- Select “exclude” by “All.”

All of the selected electrical subsystem messages will be included based upon the “Include Message” filter. All other messages will be ignored based upon the “Exclude All” filter. In this pane, all messages are captured by the second filter and thus none will be included by default.

3.3 FD Selection

The CCTK graphical applications Retriever and Measurement Monitor both require the user to select FD’s from a list. With Retriever, the selected FD’s will be retrieved from the CCTK archive subsystem. With Measurement Monitor, the selected FD’s’ current value will be displayed. A dialog box, common to both applications, is used to select FD’s from the list of

FD's currently configured in the system. This section describes the CCTK FD selection dialog box.

3.3.1 Overview

The FD selection dialog box can be called up by clicking on the toolbar button shown in Figure 3-27. Figure 3-28 shows the FD selection dialog box.

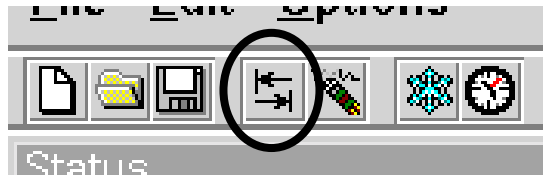


Figure 3-27. FD Selection Icon

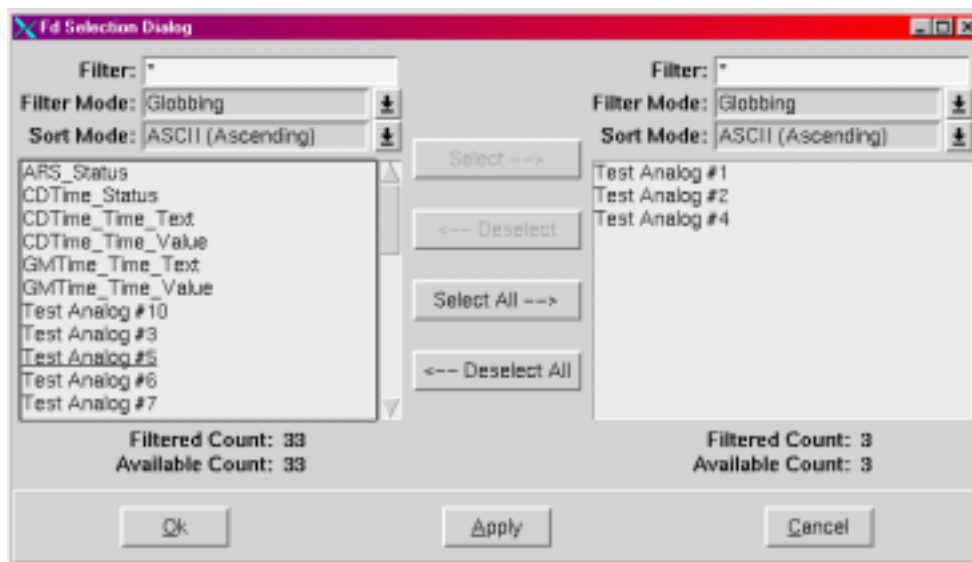


Figure 3-28. FD Selection dialog box

On the left side of the dialog box there is a list of unselected FD's and controls to operate on the unselected FD's. On the right side of the dialog there is a list of selected FD's and controls to operate on the selected FD's. In the middle of the dialog box, there are controls for moving FD's between the unselected and selected lists.

3.3.2 Selecting/Deselecting FD's

To select FD's, highlight the desired FD's within the scrolled list on the left side of the dialog. As soon as an FD is selected, the "Select -->" button in the center of the dialog box will become sensitized. Clicking this button moves the highlighted FD's from the unselected list to the selected list. This operation can be repeated as necessary until all of the desired FD's have been selected.

To unselect FD's, simply reverse the operation. Highlight the FD's in the selected list. The "<- Deselect" button in the center of the dialog box will become sensitized. Clicking this button moves the highlighted FD's from the selected list to the unselected list. Again, this operation can be repeated as necessary until all of the desired FD's have been unselected.

3.3.3 Filtering FD's

Above each FD list are two controls that allow you to filter the visible FD's. In some systems, it is common to see thousands of FD's. For this reason, there needs to be a simple, effective means by which to limit the number of FD's available for viewing.

Filtering is accomplished by selecting a filter mode and then entering an appropriate filter. There are three filter modes available:

- Globbing – simplest filtering method: use a "?" to match any single character; use a "*" to match zero or more characters.
- Regular Expression – advanced filtering method, uses the regular expression language to allow for complex matches.
- Regular Expression (ignore case) – same as regular expression except case is ignored.

Regular Expressions is a complex, powerful pattern matching language. Due to the complexity of the language, it cannot be easily explained here. Many sources are available that describe regular expressions. The FD Selection dialog box uses the Tcl regular expression engine.

For example, perform the following steps to view all FD's that start with the letter "T":

1. Select the "Globbing" filter mode.
2. Type "T*" in the Filter text entry box.
3. Hit "Enter" in the Filter text entry box.

Now, only those FD's that begin with the letter "T" will be listed in the selection list. To view all FD's that start with the letter "T" and contain the letter "A," enter "T*A*" as a filter. To return to viewing all FD's, use the "*" filter.

Below the list of FD's, two counts are displayed. Filtered Count shows the total number of FD's in the list box. If this number is large, you may want to refine the filter before searching through the list. Available Count shows the total number of FD's available. If the "*" filter is used, then Filtered Count and Available Count will always be the same value.

3.3.4 Selecting/Deselecting All FD's

Two additional buttons in the center of the dialog allow you to select and deselect all filtered FD's. The key point to remember with "Select All->" and "<- Deselect All" is that only the filtered FD's will be selected/deselected.

3.3.5 Sorting FD's

It is also possible to change how the filtered FD's are sorted within the list. The following sort modes are available:

- ASCII (Ascending)
- ASCII (Descending)
- Dictionary (Ascending)
- Dictionary (Descending)

ASCII sorts using the standard ASCII codes. Ascending starts with the lowest ASCII code; descending starts with the highest ASCII code. Dictionary sorts identically to ASCII except in two cases:

- Case is ignored except as a tiebreaker.
- If two strings contain embedded numbers, the numbers compare as integers, not characters. For example, in dictionary mode, "bigBoy" sorts between "bigbang" and "bigboy," and "x10y" sorts between "x9y" and "x11y."

3.4 Real-Time Measurement Monitoring

Measurement Monitor is used to view the current value of measurements and related information:

- Status
- FD Name
- Last Time Processed
- Raw Value
- Engineering Unit

Figure 3-29 shows an example of Measurement Monitor with several FD's displayed. Measurement Monitor is an independent window and does not display within the CCTK Client MDI interface.

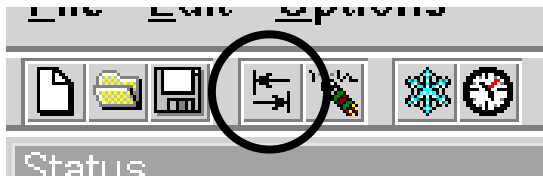


Figure 3-30. Measurement Monitor Toolbar, FD Selection Button

Select the FD's that you wish to view from the FD selection dialog. See Section 3.3 for more information on using the FD selection dialog. Once the FD's are selected, click "Ok" within the FD selection dialog. Measurement Monitor will display the selected FD's in the primary window.

3.4.2 Removing FD's

FD's can be removed from the list in one of several ways. First, the FD selection list can be opened and the FD's can be deselected through its interface. Second, all measurements can be removed by selecting the "Edit→Remove All" menu option. Third, selected measurements can be removed by selecting the "Edit→Remove Selected" menu option or clicking on the remove selected FD's toolbar button. Figure 3-31 highlights the Remove FD button on the Measurement Monitor toolbar.

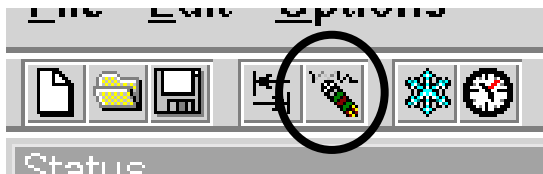


Figure 3-31. Measurement Monitor Toolbar, Removing FD's Button

3.4.3 Options

The display can be temporarily frozen by selecting the "Options→Frozen" menu item or clicking the freeze toolbar button. Figure 3-32 highlights the Freeze button on the Measurement Monitor toolbar.

The time display can be toggled between local time and GMT by selecting the "Options→Display GMT" menu item or clicking the display GMT toolbar button. Figure 3-33 highlights the display GMT button on the Measurement Monitor toolbar.

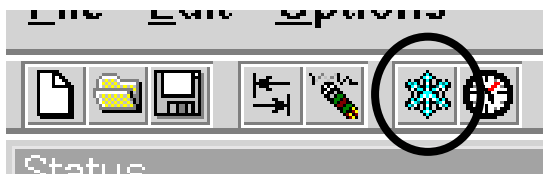


Figure 3-32. Measurement Monitor Toolbar, Freeze Button

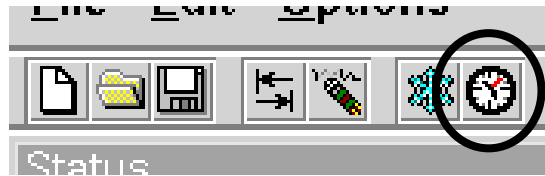


Figure 3-33. Measurement Monitor Toolbar, Display GMT Button

3.4.4 Saving and Loading Measurement Lists

It is possible to save and load lists of measurements from within Measurement Monitor. The following menu options are available:

- *New* – Clears the measurement list and resets all settings to their initial values.
- *Open ...* – Brings up a file selection dialog to select a Measurement Monitor saved file.
- *Save* – Saves the current state of Measurement Monitor; if the current file is unnamed, then a “Save As” will be performed.
- *Save As ...* – Brings up a file selection dialog to provide a name for the saved Measurement Monitor data.

3.4.5 Keyboard Accelerators

Measurement Monitor defines the keyboard accelerators shown in Table 3-4 for common operations.

Table 3-4. Measurement Monitor Keyboard Accelerators

Accelerator	Description
Ctrl+q	Exit (quit) Measurement Monitor
Ctrl+o	Open a previously saved Measurement Monitor file
Ctrl+s	Save the current Measurement Monitor file
Ctrl+f	Show the FD selection dialog
Ctrl+r	Remove the selected FD's
Ctrl+a	Show the accelerators help page
Ctrl+z	Stop the data updates.

3.5 Data Retrievals

Retriever is used to perform near real-time and historic retrievals of data. Figure 3-34 shows a screenshot of Retriever. Retriever is a stand-alone application that does not execute within the CCTK Client MDI interface.

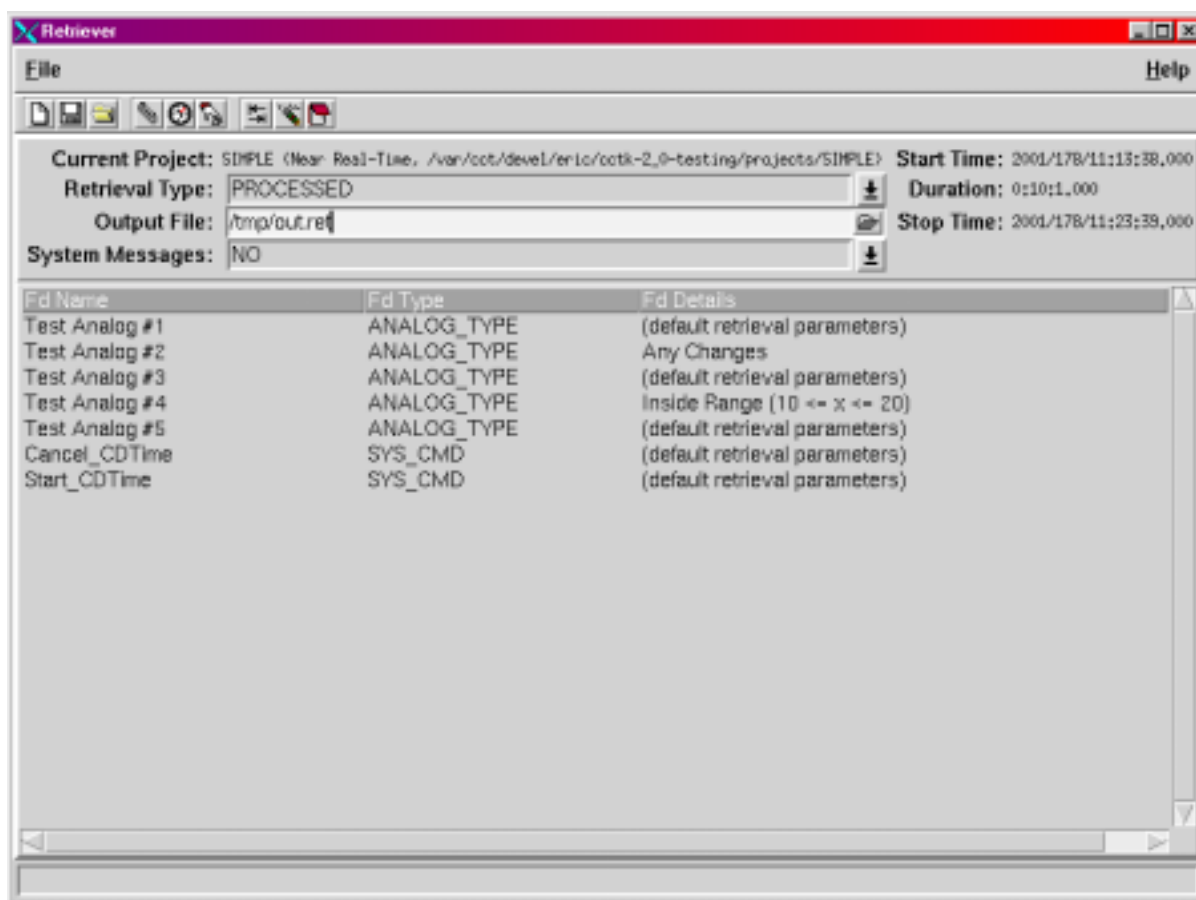


Figure 3-34. Retriever

3.5.1 Current Project

The currently selected project is shown on the main Retriever window. The project name is listed as well as if the retrieval is a near real-time retrieval or a historic retrieval. In addition, for a near real-time retrieval, the project directory is shown; for a historic retrieval, the path to the historic archive files is shown.

The project that Retriever is operating on can be changed. By clicking on the “attach” project toolbar button (the paperclip), the “Attach Project” dialog is displayed. Figure 3-35 highlights the “attach” project toolbar button. Figure 3-36 shows a screenshot of the “Attach Project” dialog.



Figure 3-35. Retriever Toolbar, Attach Project Button

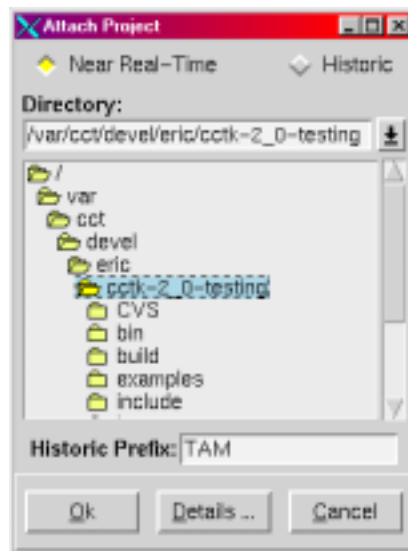


Figure 3-36. Retriever Attach Project Dialog

The “Attach Project” dialog lets you choose the type of retrieval either (near real-time or historic). If a near real-time retrieval is chosen, then the project directory should be selected in the directory browser. If a historic retrieval is chosen, then the path to the archive files is selected in the directory browser. The filename prefix for archive files is entered in the “Historic Prefix” entry widget. If you are unsure what to use for the historic prefix, see your project administrator. The default prefix “TAM” is the correct choice in most cases.

3.5.2 Retrieval Type

It is possible to select one of three types of retrievals:

- RAW – Raw measurement data is retrieved.
- PROCESSED – Processed measurement data is retrieved.
- RAW_AND_PROC – Both raw and processed measurement data is retrieved.

Select the retrieval type through the simple drop down box on the main Retriever window.

3.5.3 Retrieval Time

When performing retrieval, the time period for which the retrieval will display data needs to be specified. The current settings of start time, duration, and stop time are displayed in the main Retriever window. To modify these times, click on the “Change Time Period” toolbar button. Figure 3-37 shows the “Change Time Period” toolbar button.

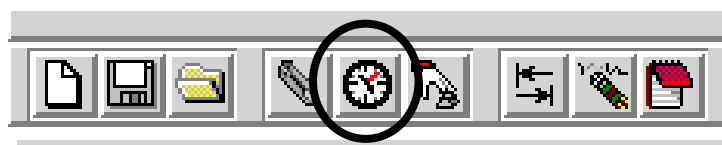


Figure 3-37. Retriever Toolbar, Change Time Period Button

Once clicked, the “Start/Stop Date/Time Entry” dialog is displayed. Figure 3-38 shows this dialog.

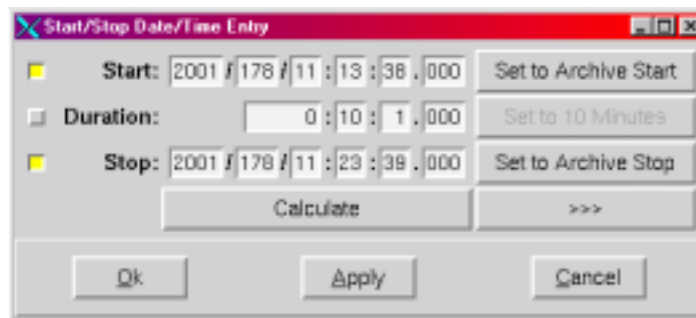


Figure 3-38. Retriever Start/Stop Date/Time Entry Dialog

This dialog is used to modify the start and stop times associated with a retrieval. There are three parameters associated with time modification: start, stop, and duration. You are able to specify any two of these parameters. The system will calculate the third. To pick which will be specified and which one will be calculated, use the checkboxes on the left side of the dialog.

Once the two parameters are selected, the detailed information for those parameters can be entered in the text entry widgets. If an invalid value is entered, the text box will turn red. For example, a year of 1500 is invalid as shown in the following figure.

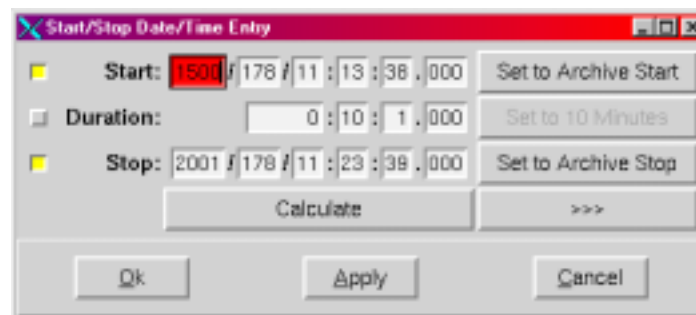


Figure 3-39. Retriever Start/Stop Date/Time Entry Dialog With Invalid Year

The calculation of the third parameter is performed automatically when Ok or Apply is clicked. To view the calculated value prior to committing the changes, use the “Calculate” button.

The controls on the right side of the dialog allow the time period to be automatically set to the archive start, 10 minutes, and archive stop time periods. These buttons are most useful with historic retrievals. They may produce undefined results when used with near real-time retrievals.

Clicking on the button with the arrows “>>>” will expand the dialog and provide additional, advanced features. Once expanded, clicking on the button with the arrows “<<<” will collapse the dialog. Figure 3-40 shows the expanded dialog.

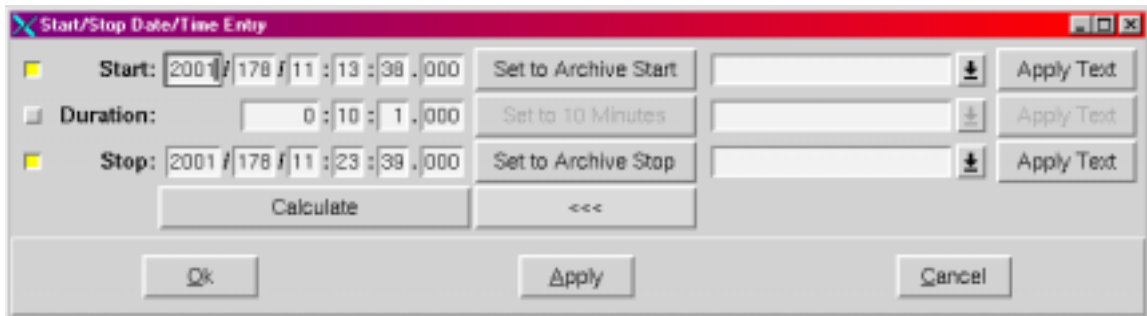


Figure 3-40. Retriever Expanded Start/Stop Date/Time Entry Dialog

The text entry boxes allow natural language text representations of time to be entered. When the apply text is clicked, the natural language text is used. For example, if you wish to retrieve data from two hours ago till now, you could enter “2 hours ago” in the first box, “now” in the bottom box, click “Apply Text” for both, and verify the times. The following natural language text is valid:

Acceptable units are year, fortnight, month, week, day, hour, minute (or min), and second (or sec). The unit can be specified as a singular or plural, as in “3 weeks.” These modifiers may also be specified: tomorrow, yesterday, today, now, last, this, next, ago.

3.5.4 Output File

Retrieval output is written to a text file. The file is specified in the “Output File” text entry widget on the main Retriever window. The file name may be typed directly into the entry widget, or it may be selected through a simple file entry dialog. The file entry dialog is accessed by clicking on the file folder on the right side of the text entry widget.

You must have adequate permission to create files in the selected directory.

3.5.5 System Messages

It is possible to retrieve system messages generated during the selected time period. A simple drop down box on the main Retriever window allows you to choose “YES” or “NO” to indicate if system messages should be retrieved. All system messages recorded during the specified time period will be retrieved.

3.5.6 FD List

The FD’s to be retrieved are listed in the lower half of the Retriever window. The following information is listed for each FD:

- FD Name
- FD Type
- FD Details

FD Type is either one of the CCTK measurement types (analog, discrete, integer, byte array) or one of the CCTK command types (system, end-item). FD Details briefly describes the retrieval details associated with that FD.

FD's can be added to the list using the FD Selection Dialog. The FD Selection Dialog is displayed when the "FD Selection" toolbar button is clicked. Figure 3-41 shows the "FD Selection" toolbar button. See Section 3.3 for more information on the "FD Selection" dialog.



Figure 3-41. Retriever Toolbar, FD Selection Button

FD's can be removed from the list in one of two ways. First, the FD Selection Dialog can be used to deselect FD's. Second, the FD's can be selected in the FD list and removed using the Remove Selected toolbar button shown in Figure 3-42.



Figure 3-42. Retriever Toolbar, Remove Selected FD Button

3.5.7 Measurement Details

For measurement FD's, it is possible to provide additional details to control the retrieval. To configure the details associated with a measurement FD, select a single measurement and click the "Modify Details" toolbar button shown in Figure 3-43.



Figure 3-43. Retriever Toolbar, Modify FD Details Button

Figure 3-44 shows the measurement details dialog box that allows specific control over the retrieval of measurement values.

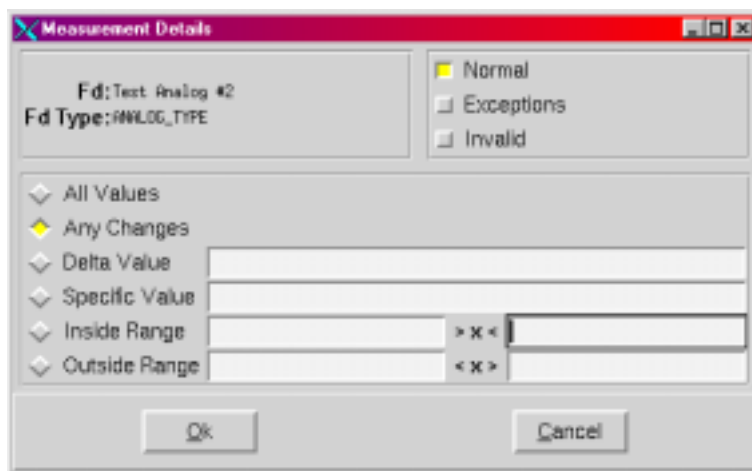


Figure 3-44. Retriever Measurement Details Dialog

In the upper left corner of the dialog, the FD name and FD type is shown. Based upon the type, different filters may not be available. In the upper right side of the screen, three checkboxes list the different classifications that can be retrieved based upon this FD. The following classifications may be retrieved:

- Normal – normal changed data.
- Exceptions – all exceptions associated with this measurement.
- Invalid – invalid processing attempts associated with this measurement.

In the lower section of the dialog, the available filters are listed. Table 3-5 lists the filters, whether or not they are available with a particular measurement type, and a short description of each.

Table 3-5. Retrieval Filters

Filter	Description
All Values	All archived values will be retrieved. This is the default. It is valid for Analog, Discrete, Integers, and Byte Arrays.
Any Changes	All archived values that differ from the previous will be retrieved. It is valid for Analog, Discrete, Integers, and Byte Arrays.
Delta Values	All archived values that differ from the previous by at least the give delta. It is valid for Analog and Integers.
Specific Values	All archived values that are equal to the given value. It is valid for Analog, Discretes, Integers, and Byte Arrays.
Inside Range	All archived values that are within the given range. It is valid for Analog and Integers.
Outside Range	All archived values that are outside of the given range. It is valid for Analog and Integers.

Only one filter may be selected for each measurement.

3.5.8 Retrieval Execution

Once all of the necessary information is entered, a retrieval can be executed by clicking the “Execute Retrieval” toolbar button shown in Figure 3-45.



Figure 3-45. Retriever Toolbar, Execute Retrieval Button

Upon executing a retrieval, the Retrieval Monitor application is started. The Retrieval Monitor application shows the progress of the retrieval and allows you some basic control over the retrieval. Figure 3-46 shows the Retrieval Monitor application.

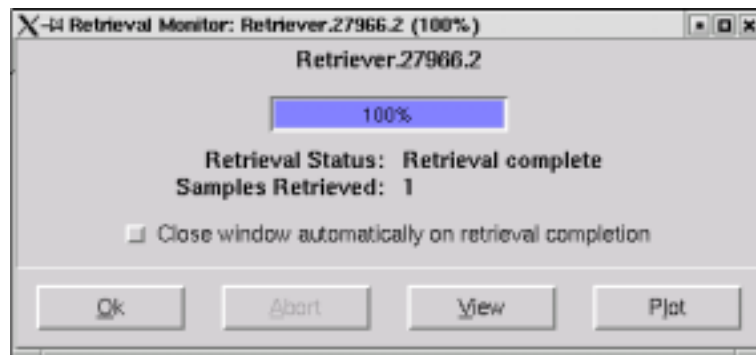


Figure 3-46. Retrieval Monitor

The progress bar shows the percentage of the Retrieval that has been completed. Retrieval status shows the current status of the Retrieval. Samples retrieved lists the number of samples retrieved thus far. The checkbox controls what Retrieval Monitor does when a retrieval completes. If it is checked, the Retrieval Monitor window will automatically close. The buttons at the bottom provide control over the retrieval. As the retrieval changes from one state to another, the appropriate buttons will become enabled.

Once a retrieval is completed, clicking on the View button will display the Retrieval output in a simple text editor and clicking the Plot button will plot the data using Grace, a 2D plotting tool. Note that only analogs, integers, and unsigned integers may be plotted. Byte arrays and discretes will be ignored by the plotting tool. Although there is no limit to the number of measurements that may be plotted on a single graph, it is recommended that you keep the number of measurements to less than 8.

Once the plot data is loaded into Grace, it is possible to use Grace to further alter and format the plot. Grace has many options for modifying the look and feel of the plot. For more information, please see the Grace documentation at <http://plasma-gate.weizmann.ac.il/Grace/>.

3.5.9 Saving and Loading Retrieval Parameter Files

It is possible to save and load retrieval parameters from within Retriever. The following menu options are available:

- New – Resets the retrieval parameters.
- Open ... – Brings up a file selection dialog to select a retrieval parameter file.
- Save – Saves the current retrieval parameters, if the current file is unnamed, then a Save As will be performed.
- Save As ... – Brings up a file selection dialog to provide a name for the saved retrieval parameter file.

3.5.10 Batch Reports

A script can be generated to initiate a series of predefined retrievals in a batch mode after system execution. Please reference the *CCTK Administrator's Manual* for more information on batch retrievals.

3.6 Virtual Strip Charts

The optional Virtual Strip Chart shown in Figure 3-47 is a dynamic two-dimensional Windows-based charting application that provides an interface for graphically viewing one or measurements of real-time data in an X-Y plot. Measurements accessed via the Selection List can be placed on one or more “tracks” for simultaneous viewing of multiple measurements. Panning and zooming allows for quick review and in-depth analysis of data trends. Any user may initiate a Virtual Strip Chart.

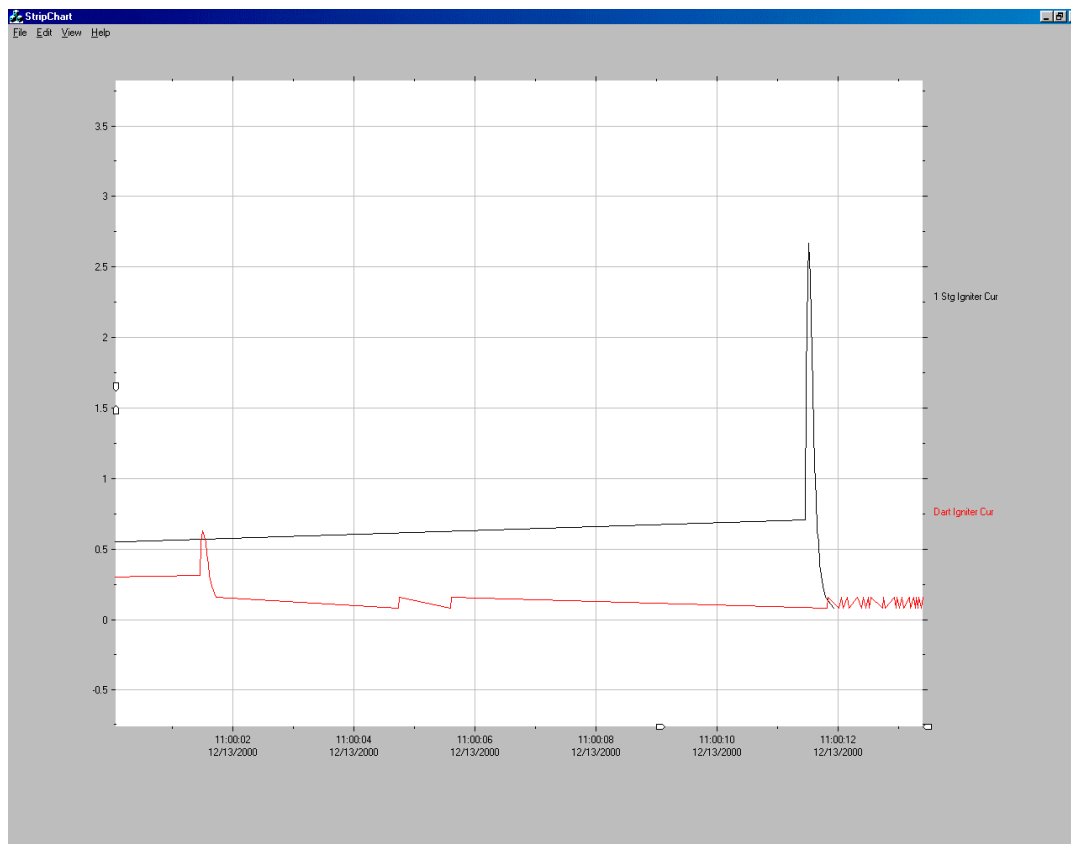


Figure 3-47. Example Strip Chart Plot

The Virtual Strip Chart can be configured vertically or horizontally. It can automatically scale using time/date or numeric counts. Data input is flexible and comprehensive with methods to add y-data, x/y data, y array data, or x/y array data. Pen colors and symbols can be modified. Property pages are available to provide for designing charts with full control of scales, captions, fonts, and tick marks.

The Virtual Strip Chart options are listed below:

Strip Chart Menu:

File

- Print (screen capture of the window)
- Exit (closes the window)

Edit

- Selection List - Show variable selection dialog box to select/deselect variables for the Strip Chart display.
- Connection - Show connection dialog box to connect to a specific data server and port address.

View

- Pause - Toggle pause/resume of the Strip Chart display. Data is still being collected while the display is paused.

- Legend - Toggle legend display on/off.
- AutoScale X - Scale the x-axis span to seconds, minutes, or hours.
- AutoScale Y - Scale the y-axis span to minimum and maximum values for the current data set.
- Properties - Show properties dialog box Figure 3-48 for detailed configuration of the Strip Chart display.

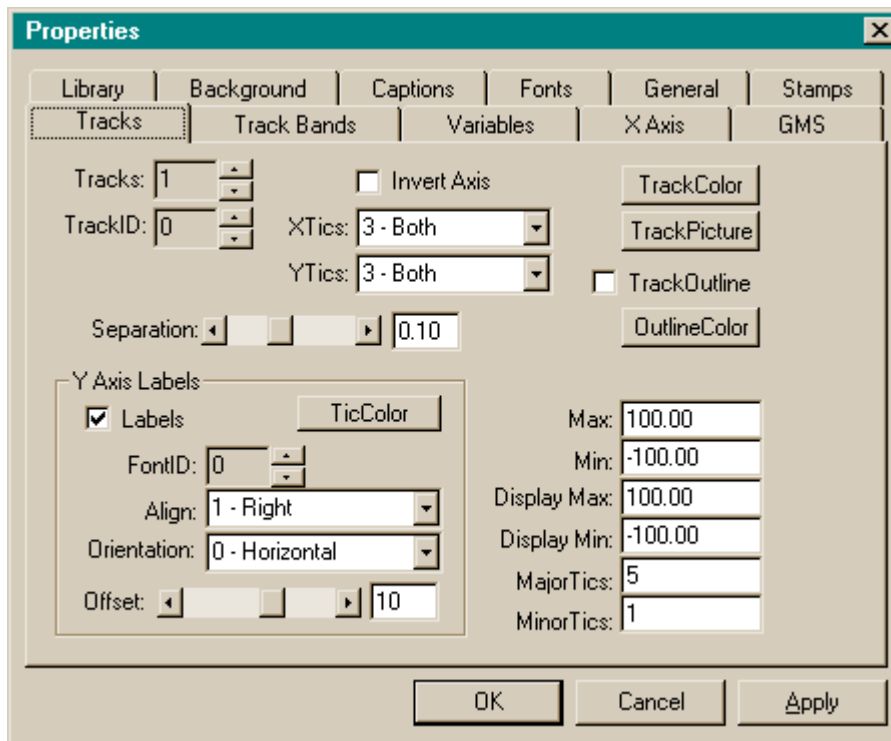


Figure 3-48. Strip Chart Properties box

Help

- About StripChart

The CCTK administrator will configure a multicast server and provide the IP port to which the strip chart client can connect.

4 CREATING DYNAMIC VISUALIZATION DISPLAYS

CCTK uses the GLG Toolkit to provide a robust dynamic visualization environment. CCTK and GLG allow complex dynamic visualizations to be created quickly and easily. No programming is necessary to create the dynamic visualizations within the CCTK environment. This codeless display environment is a key feature to the CCTK system. Three steps are involved in creating dynamic displays:

- Create the display using the GLG display builder.
- Bind data to the display by creating a simple text configuration file.
- Test the display within the CCTK system.

Each of these steps is detailed in the following sections.

4.1 Using the GLG Environment to Create a Display

GLG displays are created using the GLG display builder. The GLG display builder allows you to create complex visual representation of systems using basic constructs such as rectangles, circles, lines, and complex widgets. For information on using the GLG display builder to create visual displays, please see the GLG documentation. The documentation can be found on-line at <http://www.genlogic.com/>.

CCTK interacts with the GLG display through the GLG resource names. Each object created in the GLG display has a set of resources. These resources control color, text, and visibility. CCTK allows the binding of measurement data to any GLG resource. Therefore, CCTK measurement data can change many of the properties associated with the GLG display. GLG has many features that allow resources to be shared between objects. To use GLG to its fullest potential, review the GLG documentation on object resources.

Since CCTK measurements are linked to the GLG resources, it is important to name GLG objects appropriately during their creation. These names are used to reference the objects from within the data file. A GLG object is named from within the object properties' dialog box. Whenever naming dynamically controlled objects, be sure to give the object a name, and note that name for future reference.

Once the display has been created in the builder, save it as a file so it may be loaded by the CCTK system.

4.2 Binding Measurement Data to a Display

Once the GLG display is created using the GLG builder, data and commands can be bound to specific GLG resources. A data file is used to create these bindings. Each GLG display must have one or more data files associated with it. Multiple data files are used when a single graphical display can take on multiple roles based upon the data that it binds to. If only one data file is to be associated with a GLG display, the name of the data file is typically the same base as the GLG filename with a “.dat” extension.

The following syntax rules apply to the data file:

- White space at the beginning of lines is ignored.
- Blank lines are ignored.
- Any line that begins with a # symbol is treated as comment.
- Fields within the file are separated using commas.

All fields follow the following syntax:

```
<binding type>, <resource name>, <data string>
```

Where:

- <binding type> is one of the following “init” (for Initial Value), “meas” (for CCTK measurement), “cmd” (for CCTK command), or “exec” (for Unix commands).
- <resource name> is the name of the GLG resource the binding will apply to.
- <data string> will be interpreted as one of two things. If it is enclosed in quotes, it is a static string. If it is not enclosed in quotes, it is a CCTK descriptor reference. It is important to note that if the GLG resource is a GLG “triplet type,” then three data string values must be provided, delimited by spaces.

If the data binding is a CCTK measurement, under normal circumstances, the current value of that measurement is used. It is possible, using a special notation, to retrieve other information about the measurement (such as description, units, etc.). To retrieve other information associated with a measurement, follow the measurement name with a ‘/’ and then place one of the strings listed in Table 4-1. For example, to obtain the units for a measurement called WindSpeed, use “WindSpeed/units” as a data string.

Table 4-1. CCTK measurement data strings

String	Description
/value	The current value of the measurement (this is the default).
/description	The description associated with the measurement.
/units	The units associated with the measurement. This is only valid when the CCTK measurement is an analog or integer.
/low state	The low state associated with the measurement. This is only valid when the CCTK measurement is a discrete.
/high state	The high state associated with the measurement. This is only valid when the CCTK measurement is a discrete.
/status	The current status of the measurement.

Listing 4-1 provides a simple example of a data file.

Listing 4-1: Example GLG display data file

```
#
# Example data file, lines beginning with a # are comments
#
#
# Initial Value Bindings
#
init, Measurement/Name, FD_NAME_1/name
init, Measurement/Description, FD_NAME_1/description
init, Measurement/Units, FD_NAME_1/units
init, Measurement/Value, "-999.99"
#
# CCTK Measurement Bindings
#
meas, Measurement/Value, FD_NAME_1
meas, DisplayObject/TextString/String, FD_NAME_A
meas, Level/Value, FD_NAME_B
meas, Xform/Rotate, FD_NAME_X FD_NAME_Y FD_NAME_Z
meas, RealTimeData/Value, FD_NAME_R/forced
#
# CCTK Command Bindings
#
cmd, Push Button 1, CMD_FD_1
cmd, Push Button 2, CMD_FD_2
cmd, Push Button 3, CMD_FD_3
#
# System Command Bindings
#
exec, Button4, RunMeScript.sh arg1 arg2 arg3
```

4.3 Binding Measurement Status to a Display

This section describes how measurement status values correspond to the status on GLG displays. When displaying measurement data, it may be necessary to also indicate the status of the measurement so that the results are not misconstrued as valid data. In addition, certain measurement data may have been configured with user defined exception limits, which are commonly identified within the data display in the form of color dynamics.

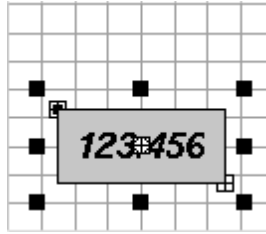


Figure 4-1. GLG text box object

The following list outlines the steps necessary to create a GLG object for displaying the current measurement status:

1. Create a text box or other suitable object (such as a filled polygon) to display the color dynamics associated with a measurements status. Set the HasResources flag to YES.
2. Edit the *Fill Color* properties of the text box.
3. Within the *Fill Color* properties dialog box, add a dynamic to the attribute property. Choose the list transformation dynamic.
4. Within the *List Transformation* dialog box, assign an appropriate resource name to the transformation's *Value Index* resource as shown in Figure 4-2 (in this example the name is "Status").
5. Edit the *List of Values* resource for the *List Transformation*, and add two additional items, or enough to make four list items in all. Each list item is representative of an individual status identifier as shown in Figure 4-2. Each integer value (0 - 3) corresponds to an appropriate status indication, or color based on the current measurement status value.
6. Now you may select individual list items and assign the appropriate color to be associated with each status value. Table 4-2 illustrates the recommended colors for the four designated status modes. Please note that any measurement status value returned ≥ 3 is categorized as critical.
7. Once the display object is created and named accordingly, you may then bind the measurement status resource to the display object's *Status* attribute within the MonCon data setup file as shown in Listing 4-2.
8. Optionally, the list transformation items may be named and constrained as reusable color elements, so that colors and effects may change dynamically and/or reused within the display for multiple instances of measurement status indicators.

Table 4-2. Measurement status and display attributes

Value	Status	Color
0	Unknown	Grey
1	Nominal	Green
2	Cautionary	Yellow
≥ 3	Critical	Red

The four status modes are derived within CCTK using the following rules: If no system or user defined measurement exception is currently being reported then the measurement's status is considered "Nominal." If the first user defined exception has been violated then the measurement's status is considered "Cautionary." If any system defined exception has been violated, or more user defined exceptions exist then the measurement's status will be shown as "Critical." Therefore, the MonCon application will report the appropriate status value (0 - 3) based on the above assumptions. It is the responsibility of the display object creator to map the appropriate colors to the status values being reported in order to produce the desired results.

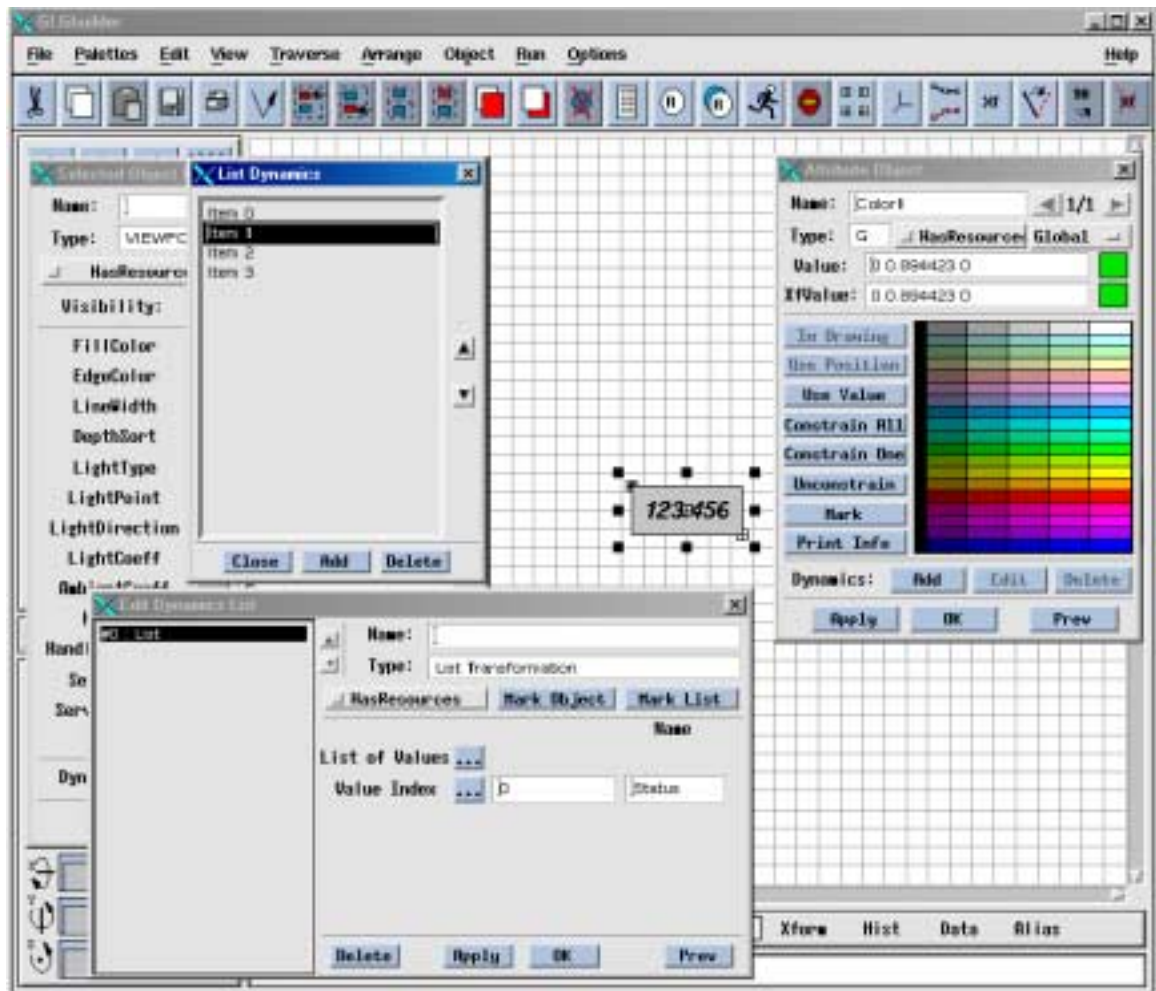


Figure 4-2. GLG builder and list transformation dialogs

Listing 4-2. MonCon binding example for measurement status resources

```
#
# Example data file binding measurement status to a display resource
#
meas, TopCircuitSensor1/Status, Dart Igniter Pwr V/status
```

An example text box widget and associated data binding file has been provided in the `${CCT_HOME}/examples/alg` directory. In addition, prefabricated text box widgets have been provided in the “widgets” subdirectory. These text box widgets are setup to utilize the measurement's status reporting scheme documented in this section.

4.4 Connecting a Display to CCTK

Once the GLG display is built and the data file is created, you may view the display using the CCTK MonCon application. For testing purposes, it is easiest to execute MonCon from the command line. A valid CCTK project must be started before MonCon is executed. To execute MonCon from the command line, perform the following steps:

- Open a shell window to the CCTK server.
- Set your KPATH environment variable to the project directory.

Run MonCon with the following command:

```
MonCon <display filename>
```

A data file with the same name as display name but with a .dat file extension must be present in the same directory as the display file. If the data file has a different name or doesn't follow this convention, then the following command may be used:

```
MonCon -i <data filename> <display filename>
```

For more details on the MonCon application and its usage, refer to the MonCon manpage in the *CCTK Administrator's Manual*.

GLOSSARY

The following terms and abbreviations are used throughout the CCTK documentation set:

Actor – Command and measurement models used in CCTK simulation.

Actor groups – Collections of actors.

Analog measurement – Measurement data that is represented as a continuously variable quantity.

ANSI – American National Standards Institute. See <http://www.ansi.org/>

API – See Application Programming Interface.

Application programming interface – A library of functions and associated data structures that allows application software to link with CCTK.

ARS – Archive and Retrieval Subsystem.

BDT – Bus Descriptor Table.

CCB – See Channel control block.

CCT – Command and Control Technologies Corporation.

CCT_HOME – Environment variable that specifies the installation directory of CCTK.

CCTK – Command and Control Toolkit.™

CCTK Client – The general reference to the program *CctkClient*, which is the main user interface for operating CCTK.

CCTKPROJECTLABEL – An environment variable that specifies an alternate term for a CCTK project.

CDB – See Configuration Database.

CDT – See Command Descriptor Table.

Channel – A channel is a means of interprocess communications. There are two types of channels, private and public. A private channel is an exclusive point-to-point connection between two processes. Private channels do not go through a distributor. A public channel is one to which any process can write.

Channel control block – An area of memory contained within the channel shared memory block that stores state information related to the channels configured in a particular instance of CCTK.

Client – Any process that requires services from another process within CCTK.

Command – A command represents a control input to an end item or operational system. The characteristics of a command are defined in command descriptions.

Command Descriptor Table – Contains information, including name and attributes, associated with system commands.

Command Response – The data sent in response to a command.

Condition Notice – A condition notice is the set of static attributes of a system message defined in the configuration database. Condition notices do not “occur.” There is no temporal aspect associated with a condition notice and they are not passed around the system. There are a set number of condition notices defined for each project. A condition notice is analogous to a “class” and a system message is analogous to an “object” of that class. When a system message is generated, an instance of a condition notice is instantiated in the system.

Configuration Database – Persistent configuration repository for all measurement parameters, command parameters, and interface specifications within CCTK.

Configuration Descriptor – A configuration record or object.

Configuration Item – A configuration item is a hardware or software entity, or an aggregation of both, which is designated for configuration management. Each configuration item has a unique set of function and/or physical attributes.

DAC – Data acquisition; see PCM data acquisition application.

Data Compression – Filtering of data to reduce bandwidth requirements.

Data Conversion – Linearization and end conversion of data.

Data Retrieval Request – Request for near real-time or permanent archived data.

Descriptor – A single named object within CCTK.

Discrete Measurement – Measurement data that is represented as one of two discrete states, such as ON or OFF.

Display Monitoring – Display monitoring is a mechanism by which a user can select and view the current value(s) of one or more measurements.

Distributor – A process that routes data through the Kernel and to external interfaces.

DMON – See Display Monitoring.

DTD – Document Type Definition, which is an XML construct that defines a document structure with a list of XML elements. See <http://www.xml101.com/> for more information.

DTDPATH – An environment variable that specifies the location of CCTK DTD files.

Dynamic Displays – Dynamic Displays associate attributes (e.g. color, blink, position, etc.) with screen objects that can, in turn, be associated with dynamic values (e.g. measurement values).

EIA – Electronic Industries Alliance.

End Item – Application device or system under the control of CCTK. The end item is a source of raw measurement data, recipient of commands, and generator of command responses.

EOF – End of file.

Exception Condition – A condition created when a measurement satisfies an event against boundary values, expected values, and masks.

Exception Event – The transition between states (positive and negative) that occurs when an expression is evaluated.

Exception Expression – An algorithm that determines the states associated with measurement values. An expression may be atomic or compound formulas consisting of operators and conjunctions.

Exception Monitoring – Generic, system provided application for monitoring exception conditions. Exception monitoring presents exception messages to the user upon detection of exception conditions detailing the conditions of the exception.

Exception Notice – An asynchronous notification of an event provided to an application.

Exception State – Persistent representation of the last exception event associated with a single measurement.

FD – See Function Designator.

Fieldbus – An open standard for industrial control I/O buses defined by ANSI/ISA-50.02, Part 2-1992.

FIFO – First in, first out.

FFI – See Frame Format Identifier.

Frame Format Identifier – An identification number within a PCM frame used to uniquely identify the contents of a frame.

Front End – Communication interfaces that connect end items to CCTK.

FSP – Frame Sync Pattern.

Function Designator – A unique symbolic name representing related data within CCTK. Function designators are defined by CCTK users and are usually associated with sensors and effectors whose value is resolved by a CCTK external interface or system application during project operations. The FD is the principal handle for data and command access within CCTK. FD's are also associated with CCTK information such as condition notices and processing chains.

GLG – Genlogic; GLG is the graphic builder and display creation tool from Genlogic Corp.

GMT – Greenwich Mean Time.

GPS – Global Positioning System.

Graphical User Interface – The interface through which clients access system functionality. This interface includes software elements (i.e. displays, controls) as well as hardware elements (i.e. workstations, input devices). Graphical and textual screen formats used to display data and accept client input.

GUI – See Graphical User Interface.

HCI – Human-computer interface; see Graphical User Interface.

Health and Status Data – Information about CCTK kernel processes that indicate of the state of the system.

HTTP – Hypertext transfer protocol.

Human-Computer Interface – See Graphical User Interface.

IBS – Interbus; IBS is a SCADA bus protocol from Phoenix Contact Corp.

Interlock Processing – The processing of a set of conditions upon which command issuance is contingent. Also referred to as “Prerequisite Control Logic.”

IP – Internet protocol.

IPC – Inter-process Communication. See UNIX system call documentation.

IRIX – The UNIX-based operating system produced by Silicon Graphics.

ISA – Instrumentation, Systems and Automation Society. See <http://www.isa.org/>.

Kernel – The core real-time services of CCTK. The kernel provides services to applications and end item interfaces for data processing, data and command distribution, and time management.

Kernel Programming Interface – An application programming interface consisting of functions provided directly by the CCTK kernel.

KMSG – Directory containing message queue information.

KPATH – An environment variable that points to the current project directory. *KPATH* is used by every CCTK process to locate the resources associated with a particular project. *KPATH* typically needs to be set if the user is going to be performing operations from the UNIX command line.

KPI – See Kernel programming interface.

KSHM – An environment variable that indicates the directory containing information on the shared memory segments created for a CCTK project. The name of the each file in the *KSHM* directory equates to the name of the shared memory segment. The contents of the file contains the shared memory key associated with the segment. Using this key, it is possible to associate the shared memory segments listed by the UNIX *ipcs* command with those used by a CCTK project.

LDT – See Link Descriptor Table.

Link Descriptor Table – An internal CCTK table that contains information associated with external commands and link records which link a measurement with an external interface.

Linked Data – A single raw measurement sample that has been acquired, time stamped, and associated with its FD handle, but not yet processed.

LSB – Least significant bit.

MDI – Multiple document interface.

Measurement – A measurement is a unit of information that is received from an end item or application. The characteristics of a measurement are defined in the CCTK configuration database. Measurements can be represented by an analog, discrete, integer, or byte array value.

Minor Frame – A minor frame is a constant sized sequence of data including a synchronization pattern for hardware identification and optionally a frame ID for software identification to uniquely identify the frame. A major frame consists of one or more minor frames.

MSB – Most significant bit.

NACK – Negative acknowledge.

NDT – Notice Descriptor Table.

NRZ – Non-return to zero.

NTP – Network Time Protocol.

Operator – A person concerned with the operation and/or administration of a control system associated with CCTK; often used interchangeably with “user.”

Packet – Any set of data associated with a packet header allowing it to be communicated through the Kernel. A packet may be the transport structure for a processed data measurement, a message, a command, a configuration descriptor, or any other type of data that circulates through the system.

PAM – Permanent Archive Media.

PCI – Personal Computer Interface. The electronic interface between Intel-class processors and peripherals.

PCM – Pulse code modulation.

PCM Data Acquisition Application (DAC) – The CCTK software program that processes PCM telemetry data.

PCMPATH – An environment variable that indicates the location of the PCM Telemetry Interface software. The PCMPATH environment variable is used by the installation script and the user environment scripts.

PDI – Payload data interleaver (a data format associated with the Space Shuttle).

Permanent Archive Media – Long-term storage for recorded data. Each PAM has a type of media and a volume. A volume is a single instance of a type of media. It may also be referred to as historical archive media. The media may be any standard storage device format, e.g., DAT or CD.

Point-to-Point Channel – A channel that is directly connected to two processes without the need for a distributor for routing the data.

POSIX – Portable operating standard for Unix. See *POSIX.4 Programming for the Real World*, O'Reilly & Associates, Sebastopol, CA, 1995.

Pseudo Function Designator – A special type of Function Designator that can be used for storage of data and to aid in communication. Typically used to designate derived data.

Qt – A Trolltech product that supports portable user interface applications; see <http://doc.trolltech.com/>.

Raw Frame – A block of data that is acquired from the end item.

Raw Measurements – Data that has not been linearized or scaled (counts).

Real-Time Control Server – The computer that hosts the CCTK server software.

Recorded Data – All measurement, command, event, message and other data that are recorded on temporary and permanent archive media for retrieval and Analysis. This includes End item Data and Configuration data.

RT – Real time.

RTCS – See Real-time control server.

RTTM – Real-time table manager.

SCADA – Supervisory control and data acquisition.

SFID – See Sub-frame identification.

SGI – Silicon Graphics, Inc.

SID – See Software Identifier.

SIGCHANSIGNAL – The CCTK signal used to indicate a packet has been placed on a channel when operating a channel asynchronously.

SIGCHANWAKEUP – An internal channel notification signal.

Simulated Universal Time – A simulated clock used by the CCTK simulation function.

Software Identifier (SID) – Internal CCTK index key for access to command measurement information. The SID is generated at configuration build time. There is one unique SID for every Function Designator (FD).

SQL – Standard Query Language.

Stale Data – Data that has not changed in value for a minimum of a user specified number of data samples.

STDARCH – The standard CCTK channel for information archival. All data sent to the STDARCH channel will be archived by the ARS.

STDDIST – Legacy CCTK channel that is no longer used.

STDMSG – The standard CCTK channel for system messages. All messages sent to the STDMSG channel will be processed and forwarded to interested processes, the system message file, and the archive subsystem.

STDRESP – Standard CCTK channel for command responses.

STDSTATUS – The standard CCTK channel for status information.

Sub-frame identification – A cyclic sequence number contained within a telemetry minor frame used by the decommutator hardware and/or software to verify major frame lock.

Super packet – A packet which contains multiple subpackets. Superpackets are used to increase efficiency by reducing overhead when transmitting data.

SUTC – See Simulated Universal Time.

System Message – An asynchronous event that notifies the user some action has occurred. System messages have a set of static attributes defined in the configuration database and a set of dynamic attributes generated at instantiation.

TAM – See Temporary archive media.

Temporary Archive – An archive that stores data in the local file system before being written to permanent archive.

Temporary Archive Media – Archive media represented by a standard UNIX file where data is stored before being written to permanent media.

TCL – Tool Command Language, a freely available platform-independent, string-based, interpreted command language. See <http://www.neosoft.com/tcl> and <http://hegel.ittc.ukans.edu/topics/tcltk/index.html> for more information.

Telemetry Data Stream – A continuous serial bit stream of time division multiplexed data.

TIX – Tk interface extension. Tix is an open open source extended widget set for Tcl/Tk. See <http://tix.mne.com/> and <http://tix.sourceforge.net/> more information.

UTC – Universal Time, Coordinated. See <http://www.time.gov/timezone.cgi?UTC/s/0/java>.

Valid – Raw and processed data values lie within a defined, valid range.

X Protocol – X Window System technology provides display and management of graphical information for UNIX-based computers. The underlying protocol is defined by X.org; see <http://www.x.org>.

XCDB – XML Configuration Database – See Configuration Database.

XIO – Extended input/out.

XML – Extensible Markup Language; a self-describing markup language defined by the World Wide Web Consortium designed to describe data. XML is an open technology; see <http://www.w3.org/XML/>.

INDEX

2	
2D	33
3	
3D	33
A	
algorithms	8, 24, 25, 28, 60
compression	25, 37
conversion	25
exception limits	26, 37
max change	25
range validity	25
stale	25, 37
analog	21, 25, 37, 74, 81
API	21, 22, 30, 31, 33, 41
application	
defined	30
development library	7
application programmer's interface	<i>See</i> API
architecture	3, 6, 7, 8, 10, 12, 19, 28, 44
archive	17
control	54
control tool	54
overview	37
archive	53
asynchronous	26, 29, 31, 41
automated sequences	32
autoscale	<i>See</i> strip chart
avionics	27
B	
bandwidth	38
batch	38, 77
bit rate	37
buffer	28
bus address	37
byte array	26
C	
C programming	7
CAUTIONARY	24

CCTK Client	
display tree	50
exiting	50
help	33
multiple document interface window	51
overview	42
status bar	52
CCTKsh	29
certification	40
channel	20, 22, 23, 34
client	4, 8, 10, 13, 22, 30, 33, 34, 37
client workstations	13
closed loop	32
color	32, 80, 83
color dynamics	82
commanding	
end item	8, 27, 28
commands	6, 19
commutation	27
compression	25
configuration	
management	33
configuration database <i>See</i> project configuration	
database	
constant	41
control loop	31
Control Shell	31
conversion	25, 37
countdown	26, 29, 55, 56, 57
commands	56
set56	
CRITICAL	24
Criticality	59, 61, 62, 63
csm	41
customization	<i>Also see</i> algorithms
interface	8
D	
data	
acquisition	8, 20, 27, 28
analysis	6, 7, 9, 10, 19, 26, 40, 77
binding to display	81
conversion	25

evaluating	25
export	10
filtering	25, 61, 65
frames	28
linked	20, 27
processing	8, 20, 24, 26, 27, 30, 31
reduction	9
retrieval overview	69
stale	25, 26
status	82
data translation documents	<i>See</i> DTD
decommutation	27
descriptor	
measurement	21
desktop	13, 30
development option	27
discrete	20, 26, 41, 74, 81
display tree	43, 50, 51, 54, 55
overview	50
domain application	6, 19, 28
DOWN	47, 50, 53
drill down	32
dsim	41
DTD	34

E

endian	24
end-item	7, 32, 74
engineering unit	37, 66
event handler	26
events	20
<i>Exceed</i>	4
exception	20, 25, 26, 31, 38, 75, 82, 84
exception limits	26
export	34, 35, 38
expression	41
external interface	6, 12, 19, 27, 28, 40

F

FD21, 22, 36, 38, 63, 64, 65, 66, 67, 68, 69, 73, 74, 75, 82	
FD selection	64, 68
Fieldbus	13
finite state machine	31
floating point	26
frame rate	37
frame structure	37
functional designator	<i>See</i> FD

G

GLG	4, 32, 33, 41, 80, 81, 82, 83, 84, 85
filename	81
Toolkit	<i>See</i> GLG
Grace	76
gsim	41

H

hardware	13, 27, 28, 37
----------------	----------------

hardware addressing	37
Health	24, 52
health and status	32
HEALTHY	24
HTTP	10

I

import	34
information hiding	32
initialization	21
integer	
signed	26
unsigned	26
interface	
external	<i>See</i> external interface
simulators	27
Internet	10, 11, 14, 15, 26
interprocess communications	20, 28

J

jitter	41
--------------	----

L

limit	26, 58, 65
limit violation	32
linearization	25, 37
local area network	10

M

master timing unit	26
MatrixX	31, 41
max change	26
MDI	42, 43, 51, 52
MDI window	51
Measurement Monitor	21, 29, 63, 66, 67, 68, 69
tool	66
Message ID	61
messaging	8, 28
Microsoft Access	35
middleware services	6, 19
missing element modeling	8, 9, 40
mission elapsed time	26
mode	34, 48, 58, 65, 66, 77, 83, 84
initial	48
MonCon	33, 83, 84, 85
monitoring measurements	
measurements	<i>See</i> Measurement Monitor

N

Network Time Protocol	26
notices	22, 26, 36, 37
notification	26

O

Oracle	35
--------------	----

P

PCI	13, 28
-----------	--------

peer-to-peer	10
performance	10, 13, 16, 20, 22, 41
profiles	18
permission	30, 73
Platforms	13
playback	9
plotting	76
polynomial	24
port	21, 37
port address	37, 78
POSIX	12, 13
processed value	25, 26
processing algorithms	<i>See</i> algorithm
program	
application	7, 9, 28
project	
administrator	21, 37, 54, 55
configuration	34
configuration database	9, 37
configuration management	6, 19
forced stop	49
mode	<i>See</i> mode
opening	44
preview	46
shutdown	<i>See</i> project stopping
starting	47
states defined	47
stopping	49
project configuration	
file	34
project manager	34
project_config.tcm1	46
protocol	10, 27, 28, 37

Q

quick look	8
------------------	---

R

ramp	41
random	41
range	4, 10, 25, 26, 41
RangeNet	4, 9
raw	25, 27, 28, 71
real-time	
application example	31
control server	<i>See</i> RTCS
kernel	6, 12, 19, 20, 24, 41
tables	20, 21
recording	<i>See</i> retrieval
regular expression	65
retrieval	9, 10, 37, 38, 69, 70, 71, 72, 73, 74, 76, 77
batch	10, 77
filters	75
historical	9, 10, 26, 29
near real-time	9, 29, 69, 70, 71, 72
overview	37
parameters, saving	77
scripts	30, 71, 72, 77

retriever	29, 38, 63, 69, 70, 71, 72, 73, 74, 75, 76, 77
tool	69
RTCS	13, 30, 33

S

sample rate	37
saw-tooth	41
SCADA	27
schematic	32
scripting	7, 8, 29, 41
segment	89
server	8, 10, 13, 37, 78, 85
shared memory	21, 22
shared memory segments	89
shutdown	<i>See</i> project stopping
forced	<i>See</i> project forced stop
SID	22
simulation	7, 8, 28, 29, 31, 33, 34, 40, 41
simulation tool	
defined	41
sine	41
SL-GMS	32
square	41
stale	25
STARTING_UP	48, 53
STARTUP_FAILED	49
state	23
state history	48, 49, 53, 54
clearing	54
tool	53
status23, 25, 26, 28, 34, 43, 44, 45, 51, 52, 54, 55, 56, 76, 82, 83, 84, 85	
measurement display	82
status bar	43, 44, 52
strip chart	8, 29, 77, 78, 79
overview	77
support utilities	6, 19
Sybase	35
symmetric multi-processing	13
sync patterns	37
system	
applications	29
system identifier	<i>See</i> SID
system message	9, 20, 29, 37, 58, 61, 62, 63, 73
example	63
filters	31, 38, 58, 60, 61, 62, 75
retrieving	73
tool	58

T

tables	<i>See</i> real-time tables
Tcl	7, 8, 29, 41
TCP/IP	10, 13
telemetry	26, 27, 28
testing	34, 40, 85
theory of operation	6
time	
acquisition	8

control.....	29, 53, 55, 56, 57
control tool	55
distribution.....	6, 8, 19, 20
GPS.....	26
internal.....	22, 26
IRIG-B.....	26
shift.....	26
stamp	26
synchorize.....	26
synchronization.....	8, 26
system.....	26
time control	56
timers.....	8, 26
toolbar	
CCTK Client.....	43
topology.....	10
training	8, 9, 40
trigger	26, 31
T-Zero.....	4, 9

U

universal time	<i>See</i> system time
UP.....	24, 49, 53
user	

access.....	30
accounts.....	9
applications	30
logon.....	30
notes	48, 49, 53
user display	
creation.....	33

V

visualization	8, 32, 33, 80
VME.....	13, 28

W

window	
management	51
Windows	4, 13, 51, 52, 77
World Wide Web	<i>See</i> Internet

X

X client emulation.....	13
X client/server	10
x/y data.....	<i>See</i> strip chart
XML.....	7, 8, 21, 34, 35, 36, 46, 54